AUTOMATICĂ • MANAGEMENT • CALCULATOARE

amc 51

Fl. Ciocîrlie, P. Eleş

PASCAL ŞI PASCAL CONCURENT
(Ciclul „Limbaje de programare")

J. Bérka (RPU), S. Keresztély (RPU)
M. Sîrbu, A. Davidoviciu
Z. Benyó (RPU)
L. Orăşanu, Fl. Filip
M. Muratcea, A. Varga
M. Dumitru, A. Moangă
A. Donciulescu, C. Vasiliu
A. Alexandru, L. Iacob
P. Rădulescu, O. Gheorghiu
T. Călin, I. Dima
V. Vişan, V. Stănescu

„O PUNTE ÎNTRE ŞTIINŢĂ ŞI TEHNOLOGIE"
(Congresul Mondial al Federaţiei Internaţionale de Automatizare IFAC '84)

- Conducerea cu microcalculatoare
- Sisteme mari • PAC • Educaţie
- Sisteme expert şi om-maşină
- Decizii ierarhice şi multicriteriale
- Transporturi • Aplicaţii spaţiale
- Hidrometeorologie, calitatea aerului, gospodărirea apelor

T. Geber, I. Dobre
C. Busuioc, D. Horhoianu
M. Niţă, G. Olaru
M. Stănciulescu

TESTĂRI AUTOMATE PENTRU MINICALCULATOARELE CORAL ŞI INDEPENDENT
(Ciclul „Service pentru calculatoare")

N. Patrubany, N. Pop
A. Kiss, N. Socaciu
D. Szász

MICROCALCULATOARELE PERSONALE PRAE ŞI LIMBAJUL LOR BASIC
(Ciclul „Calculatoare personale")

V. Sima, Th. Popescu

PORTABILITATEA PROGRAMELOR
MODELAREA SERIILOR DE TIMP
(Ciclul „Algoritmi şi pachete de programe")

Th. Popescu, V. Sima
A. Varga, N. Ivan

PROIECTAREA ASISTATĂ DE CALCULATOR (PAC)

- Estimarea recursivă
- Pachete de programe PAC

SERIE CONTINUĂ DE SINTEZE, CERCETĂRI APLICATIVE, INSTRUIRE, INFORMARE, ÎN SISTEME AUTOMATE, INFORMATICE, ELECTRONICE, DE CONDUCERE

Vol. 51

AUTOMATICĂ MANAGEMENT CALCULATOARE

- Ciclul special „Congresul Mondial IFAC '84"
- Limbajele PASCAL și PASCAL CONCURENT
- Testări automate pentru minicalculatoare
- Microcalculatoarele personale PRAE și limbajul lor BASIC
- Algoritmi și pachete de programe
- Proiectarea asistată de calculator

EDITURA TEHNICĂ
BUCUREȘTI – 1985

a
m
c

Vol. 51

Traduceri, adaptări : ARISTIDE PREDOI, ADRIAN DAVIDOVICIU, MARCEL SÎRBU, DAN DOBRESCU, SANDU LAZĂR, ALEXANDRA TATU, MATEI CONSTANTIN, VASILE SIMA, RADU MAGDA, DOINA TUDOR, NICOLAE TUDORANCESCU, ILIE PAIUȘI, CAMELIA BASARAB, DAN RĂDULESCU, VIRGIL NICULA, ȘERBAN SPOREA, RADU MANOLESCU, NICOLAE ALGIU, LEONARD STOIAN HOROBEȚ, GABRIEL SPIRIDON, EUST. STANCIU, ȘTEFAN LUPESCU, OLEG CRĂMĂRĂSCU, AURELIAN SIMIONESCU.

Recenzii : SANDU LAZĂR, ADRIAN DAVIDOVICIU, LIVIU DUMITRAȘCU, GORUN MANOLESCU, TOMA GEBER, ADRIAN PETRESCU, HORIA CIOCÂRLIE, PETRU ELEȘ, PAUL CONSTANTINESCU.

Colectivul de traducere, adaptare, recenzare menționat, a lucrat la toate volumele AMC 48—51.

Redactori : ing. **PAUL ZAMFIRESCU**
ing. **MIRCEA GROSU**

Tehnoredactor : OLIMPIADA NISTOR
Coperta : SIMONA DUMITRESCU

Bun de tipar : 16. sept. 1985. Coli tipar 22,5

Tiparul executat la
Intreprinderea Poligrafică „Banatul"
Timișoara

Vol. 51

CUPRINS

a
m
c

Vol. 51

În paginile 264 și 360 sînt cuprinse informații asupra unor manifestări din țară și străinătate, ca și recomandări de cărți.

Ciclul

„Limbaje de programare"

LIMBAJELE PASCAL ŞI PASCAL CONCURENT. SINTEZĂ

Horia Ciocârlie, Petru Eleş

Institutul Politehnic Timişoara

Conceptul de programare structurată a constituit punctul de plecare al dezvoltării programării calculatoarelor pe fundamente ştiinţifice. Pe baza lui, N. Wirth a definit în anul 1971 limbajul PASCAL [1]. Pornind de la PASCAL P.B. Hansen a reuşit să introducă într-un limbaj evoluat conceptele programării concurente, definind în anul 1975 PASCALUL CONCURENT [2].

În lucrare se face o prezentare a limbajelor de programare PASCAL şi PASCAL CONCURENT, din punctul de vedere al conceptelor fundamentale care au stat la baza definirii lor, şi anume:

— structurarea şi modularizarea acţiunilor şi datelor unui program;

— programarea concurentă în limbaje de nivel înalt.

Maniera de prezentare se bazează pe întrepătrunderea unor noţiuni mai generale de programare, cu aspecte practice specifice realizării lor în cele două limbaje.

1. Structurare şi concurenţă în programare

Limbajul PASCAL este un rezultat al evoluţiei conceptelor apărute ca urmare a crizei ce caracteriza domeniul programării calculatoarelor la sfîrşitul anilor '60. În această perioadă răspîndirea pe plan mondial a prelucrării datelor cu calculatorul a cunoscut un salt deosebit. S-au putut astfel aborda şi rezolva probleme din ce în ce mai complexe. Programele mari corespunzătoare acestor probleme s-au complicat în aşa măsură încît au devenit o „junglă" de nepătruns, chiar şi pentru autorii lor. Înţelegerea, depanarea şi modificarea unor astfel de programe prezenta adesea dificultăţi deosebite. Limbajele de programare existente [3] şi absenţa unor principii clare care să impună o „disciplină a programării", au favorizat în mare măsură această stare a lucrurilor. A apărut în mod natural necesitatea elaborării unei metodologii generale care să permită scrierea în mod sistematic a unor programe eficiente în realizare, exploatare şi întreţinere.

Ca urmare a acestei necesităţi s-a cristalizat *metoda proiectării şi programării structurate* [4, 5]. Un program structurat este construit din unităţi funcţionale

bine conturate, ierarhizate conform naturii intrinseci a problemei. În interiorul unor astfel de unități structurarea se manifestă atît la nivelul acțiunilor (instrucțiunilor), cît și al datelor prelucrate. Rezultă programe clare, ordonate, inteligibile, fără salturi și reveniri.

Programarea structurată este o metodă independentă de limbajul de programare, ea acționînd la nivelul stilului de lucru. Totuși, practica a demonstrat că limbajul poate înlesni în mod hotărîtor realizarea dezideratelor enunțate mai sus. Aceasta este valabil prin excelență pentru limbajul PASCAL, proiectat în mod sistematic, conform conceptelor științifice ale programării structurate. În ideea simplificării și ordonării activității de programare, PASCAL facilitează și chiar impune scrierea unor programe clare, ușor de întreținut, depanat și modificat [6]. Datorită acestor calități, limbajul s-a răspîndit rapid în toate domeniile în care se folosește calculatorul, fiind astăzi unul dintre cele mai cunoscute și utilizate limbaje pe plan mondial.

În domenii speciale în care practica a demonstrat necesitatea introducerii unor facilități specifice pentru ca un limbaj de programare să devină util, PASCAL a constituit un punct de plecare preferat pentru definirea unor noi limbaje.

Astfel, în domeniul programării de sistem, PASCAL nu a permis abordarea unor categorii de probleme specifice: multiprogramarea, sincronizarea unor procese care folosesc resurse comune, protecția resurselor și datelor nepartajabile etc. A rămas deschisă problema găsirii unei metodologii cu ajutorul căreia să se scrie în mod eficient programe concurente inteligibile, modificabile și fiabile. Prin definirea PASCALULUI CONCURENT s-a demonstrat că un limbaj de nivel înalt adecvat poate să constituie unealta corespunzătoare pentru scrierea unor astfel de programe. PASCAL CONCURENT păstrează spiritul și forma limbajului PASCAL [6]. Pe această bază s-au introdus însă noi concepte menite să rezolve probleme specifice programelor de tip sistem și care au dat limbajului un caracter cu totul deosebit. Domeniul exclusiv al programării de sistem a devenit astfel accesibil unui cerc tot mai larg de programatori, permițînd totodată creșterea spectaculoasă a productivității elaborării programelor concurente.

Problemele prezentate mai sus au constituit motivația principală datorită căreia un colectiv de specialitate de la Institutul Politehnic „Traian Vuia" Timișoara s-a preocupat în mod constant în ultimii ani de diferite aspecte teoretice și practice vizînd limbajele PASCAL și PASCAL CONCURENT. S-au realizat astfel, compilatoare PASCAL și PASCAL CONCURENT pentru calculatoarele Felix C (256, 512, 1024), ca și implementări ale celor două limbaje pe calculatoarele din familiile Independent și Felix M-18 [7, 8, 9, 10].

2. Structuri de acțiuni și de date

2.1. Instrucțiuni

Gruparea acțiunilor unui program în structuri [11] conferă acestuia suplețe, claritate și flexibilitate. Limbajul PASCAL permite descrierea acțiunilor ca niște *instrucțiuni simple*, asignarea (atribuirea) și apelul de procedură, sau sub

forma unor structuri de acţiuni cu o intrare şi o ieşire numite *instrucţiuni structurate*. Ele provoacă executarea secvenţială, repetată sau condiţionată,a uneia sau mai multor instrucţiuni şi se împart în mod corespunzător în trei categorii :

a) **Instrucţiunea compusă,** de forma :

begin instrucţiune 1 ; instrucţiune 2 ; ... instrucţiune n *end*

Efectul ei este executarea instrucţiunilor componente (instrucţiune 1 ... instrucţiune n) în ordinea strictă în care au fost scrise.

b) **Instrucţiuni repetitive**

— Instrucţiunea *while*, de forma :

while condiţie *do* instrucţiune

Se realizează executarea repetată a instrucţiunii care urmează cuvîntului cheie *do*, atîta timp cît condiţia este adevărată.

— Instrucţiunea *repeat*, de forma :

repeat instrucţiune 1 ; instrucţiune 2 ; ... instrucţiune n *until* condiţie

Instrucţiunea provoacă repetarea executării listei de instrucţiuni dintre cuvintele cheie *repeat* şi *until*, pînă cînd condiţia devine adevărată.

— Instrucţiunea *for* cu următoarele două forme posibile :

for contor : = expresie1 *to* expresie2 *do* instrucţiune
for contor : = expresie1 *downto* expresie 2 *do* instrucţiune

Instrucţiunea *for* provoacă executarea instrucţiunii care urmează cuvîntului cheie *do*, în mod repetat pentru fiecare valoare a contorului. Acesta primeşte iniţial valoarea obţinută prin calcularea expresiei 1 şi se incrementează (forma cu *to*) respectiv se decrementează (forma cu *downto*) cu pasul 1, la fiecare iteraţie pînă la valoarea expresiei 2.

c) **Instrucţiuni condiţionale**

— Instrucţiunea *if* cu următoarele două forme posibile :

if condiţie *then* instrucţiune 1 *else* instrucţiune 2
if condiţie *then* instrucţiune

În prima formă, în funcţie de condiţie se selectează o instrucţiune din cele două posibile. Forma fără *else* provoacă executarea condiţionată a instrucţiunii care urmează cuvîntului cheie *then*.

— Instrucţiunea *case*, de forma :

case expresie *of*
constantă 1 : instrucţiune 1 ;
constantă 2, constantă 3 : instrucţiune 2 ;
....
constantă n : instrucţiune m
end

Instrucţiunea are ca efect evaluarea expresiei care urmează cuvîntului cheie *case* şi executarea doar a instrucţiunii a cărei etichetă coincide cu valoarea obţinută.

2.2 Date

2.2.1. Noţiunea de tip

Instrucţiunile unui program acţionează asupra datelor sale conform algoritmului corespunzător problemei ce urmează a fi rezolvată. În ceea ce priveşte modalităţile de limbaj create pentru descrierea datelor, PASCAL a prezentat un salt cu totul excepţional, care a influenţat în mod decisiv evoluţia ulterioară a limbajelor de programare.

La baza descrierii datelor în PASCAL stă noţiunea de *tip*. Tipul unei mărimi (constantă, variabilă, valoarea unei expresii etc.) determină mulţimea valorilor pe care aceasta le poate lua, precum şi setul de operaţii ce i se pot aplica. Descrierea datelor în PASCAL presupune definirea de către programator a unor tipuri de forma dorită. Limbajul permite descrierea în primă instanţă a unor tipuri simple (nestructurate). Acestea constituie cărămizile de bază, pornind de la care se descriu structuri formate din componente de diferite tipuri, legate într-o construcţie de tip nou. Mijloacele de structurare ale limbajului (tabloul, articolul, mulţimea etc.), permit astfel descrierea unor tipuri de date de orice complexitate [12].

Tipul, după cum s-a arătat, determină o mulţime de *valori posibile*. Entităţile care pe parcursul executării programului pot lua diferite valori de un anumit tip se numesc *variabile*. O variabilă se declară, înainte de a fi utilizată, ocazie cu care i se asociază un identificator şi i se specifică tipul, ca în exemplul de mai jos:

var identificator : tip

2.2.2. Tipuri simple

În limbajul PASCAL există patru tipuri simple standard care pot fi utilizate în program fără a fi definite. De asemenea, programatorul poate defini tipuri simple proprii. O proprietate comună a tipurilor simple este ordonarea mulţimii valorilor corespunzătoarelor (ele se mai numesc şi tipuri scalare).

2.2.2.1. *Tipurile standard.* a) Tipul boolean este format din mulţimea valorilor logice false şi true între care există relaţia de ordine false< <true.

b) Tipul integer este o submulţime a mulţimii numerelor întregi, care depinde de implementare.

c) Tipul real este format dintr-o submulţime a numerelor raţionale, dependentă de implementare, reprezentînd aproximaţii ale numerelor reale.

d) Tipul char este format din mulţimea caracterelor tipăribile (deci este şi ea dependentă de implementare). Constantele de tip caracter se reprezintă prin caracterul respectiv cuprins între apostrofuri: 'A', 'x'.

2.2.2.2. *Tipul enumerare.* Mulţimea valorilor unui tip enumerare se defineşte prin enumerarea identificatorilor ce reprezintă aceste valori, ca în exemplele de mai jos:

```
type zilucru = (luni, marţi, miercuri, joi, vineri);
     culori  = (roşu, galben, albastru);
     grad    = (soldat, fruntaş, caporal, sergent);
     boolean = (false, true);
```

Ordonarea în mulţimea valorilor unui tip enumerare este determinată de poziţia identificatorilor în listă [7].

Identificatorii reprezentînd valori ale mărimilor de tip enumerare („luni", „roşu", „fruntaş") pot fi folosiţi în calitate de constante, ceea ce măreşte considerabil claritatea programului.

Exemplu :

```
var zi : zilucru ;  în loc de      var zi : integer ;
.............                      ...........................
zi : = marţi ;                     zi : =1 ;
.............                      ..........
f zi< >vineri then .....           if zi< >4 then .....
```

2.2.2.3. *Tipul subdomeniu.* În anumite aplicaţii se cunosc limitele între care o variabilă de tip scalar poate lua valori. În acest caz tipul variabilei se poate defini ca un subdomeniu al tipului scalar respectiv. În definiţia tipului se indică limitele inferioară şi superioară ale intervalului, ca mai jos :

```
type gradat  = fruntaş .. sergent ;
     ti      =20..115 ;
     cifre   ='0'...'9' ;
```

Nu se definesc subdomenii ale tipului real.

2.2.3. Tipuri structurate

2.2.3.1. *Tabloul.* Tabloul este o structură formată dintr-un număr fix de componente, toate de acelaşi tip. Componentele unui tablou se numesc elemente ; selectarea unui element de tablou se face pornind de la numele variabilei tablou şi indicînd poziţia elementului în cadrul structurii prin aşa-numitul indice [7].

La definirea unui tablou se precizează atît tipul elementelor (tipul de bază al tabloului), cît şi al indicelui. Prin acesta din urmă se fixează implicit şi numărul componentelor tabloului.

Exemple :

```
type t = array (.ti.) of tb
     ti : tipul indicelui, care poate fi orice tip scalar, mai puţin real ;
     tb : tipul de bază, cu privire la care nu există nici o restricţie.
type orelucru = array (.zilucru.) of real ;
     text     = array (.1..20.) of char ;
     matrice  = array (.1..10.) of (.3..15). of integer ;
sau  matrice  = array (.1..10,3..15.) of integer ;
var ore : orelucru ; m : matrice ; i : integer ;
..............................................................
     ore (.marţi.) : =8.25 ;
     m (.I+1,4.) : =7 ;
```

Indicele este o expresie a cărui tip trebuie să corespundă cu cel indicat la definirea tabloului (ceea ce presupune implicit şi încadrarea valorii expresiei în limitele impuse).

2.2.3.2. *Articolul*. Articolul reprezintă o reuniune a unui număr fix de componente, care pot fi de tip diferit. El este astfel o metodă de structurare mai generală decît tabloul. Componentele articolului se numesc cîmpuri. La definirea articolului se specifică atît numele cîmpurilor, cît şi tipul lor [7].

Exemple :

```
type data =record
          zi : 1..31 ;
          luna : 1..12 ;
          an : 1900..2000
          end ;
     persoana = record
          nume : array (.1..10.) of char ;
          datan : data
          end ;
```

Selectarea cîmpurilor se realizează indicînd atît identificatorul variabilei articol, cît şi al cîmpului, printr-un procedeu numit calificare, ca în exemplul de mai jos :

```
var pers : persoana ;
..................
pers. nume : ='popescu' ;
pers. datan.an : =1950 ;
```

Structuri de articol diferite, dar care prezintă componente cu semnificaţie înrudită, sau chiar porţiuni identice, pot fi descrise ca variante ale aceluiaşi tip articol. În acest scop, la definirea tipului se va preciza un cîmp, numit cîmp selector, a cărui valoare momentană va indica structura curentă a articolului. Pentru diferitele valori posibile ale cîmpului selector se vor descrie cîmpurile corespunzătoare variantelor.

Exemplu :

```
type sex = (m, f) ;
     persoana 2 = record
          nume : array (.1..10.) of char ;
          case sexul : sex of
          m : (datan : data ; greutate : real) ;
          f : (nrcopii : integer)
          end ;
```

Cîmpul selector se numeşte „sexul" şi este de tip „sex". Cele două structuri posibile ale unui articol de tip „persoana 2" sînt prezentate în fig. 1.

2.2.3.3. *Mulţimea*. În limbajul PASCAL este posibil ca elementele mulţimii de valori, corespunzătoare unui tip să fie la rîndul lor mulţimi. Acest caz corespunde definirii unui aşa-numit tip mulţime, ca în exemplul de mai jos :

```
type t =set of t0
```

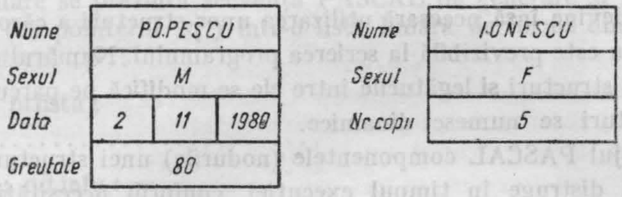


Fig. 1. Variantele articolului de tip „persoana 2".

Valorile posibile pentru o variabilă de tipul „t" sînt toate submulţimile mulţimii valorilor corespunzătoare tipului „t0", inclusiv mulţimea vidă.

Dacă definim t0 = (a, b, c), o variabilă de tipul „t" de mai sus ar putea lua ca valori una din următoarele mulţimi:

{ } ; {a} ; {b} ; {c} ; {a, b} ; {a, c} ; {b, c} ; {a, b, c}.

Între mulţimi astfel definite se pot executa, în sensul matematic cunoscut, operaţiile de reuniune, intersecţie, scădere (*or*, *and*,—) şi se pot aplica operatorii relaţionali egal, diferit, incluziune în ambele sensuri (=, < >, < =, > =). De asemenea se poate verifica apartenenţa unui element la o mulţime cu ajutorul operatorului relaţional *in* [1].

2.2.3.4. *Fişierul* Fişierul este o mulţime organizată de informaţii. Aceste informaţii sînt grupate sub formă de înregistrări, care în marea majoritate a cazurilor sînt stocate pe un suport extern. Numărul înregistrărilor este neprecizat şi din acest punct de vedere fişierul poate fi considerat ca o structură dinamică (v. § 2.2.4).

Una din caracteristicile esenţiale ale unui fişier este tipul înregistrărilor sale (acelaşi pentru întregul fişier). Acesta se numeşte tipul de bază al fişierului şi se indică la definirea tipului fişier, astfel:

type tipfiş = *file of* tipbază

O variabilă de tip fişier reprezintă o structură organizată conform tipului de bază, prin intermediul căreia se execută prelucrările asupra fişierului (este practic o zonă tampon). Ea conţine la un moment dat o înregistrare a fişierului.

Definiţia de tip *file* nu descrie caracteristicile dependente de suport şi de organizare ale fişierului propriu-zis. Acestea se indică prin alte construcţii de limbaj, ale căror formă şi semnificaţie depind de implementarea pe un anumit sistem de calcul [8].

Prelucrarea fişierelor se execută prin rutine standard pentru deschidere, închidere, citire, scriere, punere la zi etc.

2.2.4. Structuri de date dinamice. Tipul pointer

Tipurile tablou, articol, mulţime descriu, aşa cum s-a arătat, structuri de formă şi dimensiune fixă cunoscute la compilarea programului. Ele reprezintă din acest motiv structuri statice. Fişierul, chiar dacă în ansamblu este o structură de dimensiuni nelimitate, poate fi privit ca structură statică, prin prisma semnificaţiei asociate unei variabile de tip fişier.

Deseori devine însă necesară utilizarea unor structuri a căror formă sau/şi dimensiune nu este previzibilă la scrierea programului. Numărul componentelor unei astfel de structuri şi legăturile între ele se modifică pe parcursul execuţiei. Aceste structuri se numesc dinamice.

În limbajul PASCAL componentele (nodurile) unei structuri dinamice se pot crea sau distruge în timpul execuţiei, conform necesităţilor exprimate dinamic în program. În acest scop există posibilitatea de a se defini un tip special, denumit pointer, ca în exemplul de mai jos:

type tp=∂t;

Valorile corespunzătoare tipului „tp" sînt adrese care permit accesul la variabile alocate dinamic, de tipul „t". În fig. 2 se prezintă două structuri dinamice, o listă liniară şi respectiv un arbore. Descrierile în PASCAL ale nodurilor corespunzătoare acestor structuri sînt:

```
type ptlista = ∂ nodlista;
     nodlista = record
                  info : tipinfo;
                  urm : ptlista
                end;

type ptarbore = ∂ nodarbore;
     nodarbore = record
                   info : tipinfo;
                   stîng, drept : ptarbore
                 end;
```

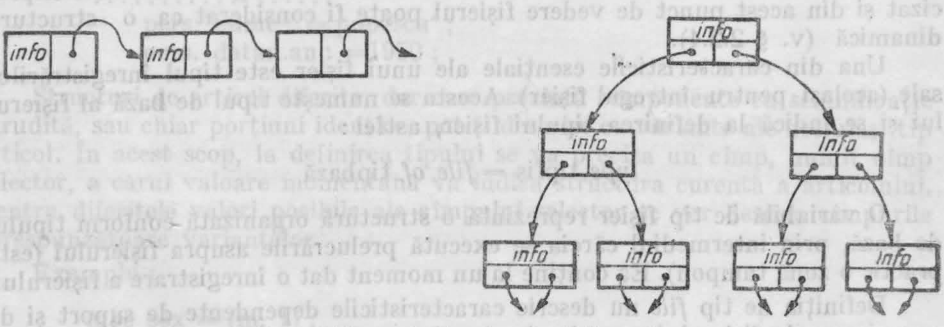


Fig. 2. Exemple de structuri dinamice.

Alocarea dinamică a unei variabile de tip t se realizează la cererea explicită din program, prin apelul procedurii standard new sub forma: new(p); „p" este o variabilă de tipul tp=∂t.

În afara alocării zonei de memorie corespunzătoare structurii de tip „t", apelul lui new efectuează şi asignarea variabilei „p" cu valoarea adresei structurii respective. Toate referirile la variabila alocată dinamic se realizează prin pointerul „p", sub forma: p∂.

Procesul invers, de eliberare a unei zone alocate dinamic, referite prin pointerul p, se execută ca urmare a apelului procedurii standard dispose sub forma: dispose (p).

În continuare se prezintă secvenţa PASCAL de generare şi introducere a unui nou nod (cu pointerul „q") într-o listă liniară de forma din fig. 2, după nodul cu pointerul „p" :

```
var p, q : ptlista ;
.................................
n e w (q) ; q∂.info : = . . .
q∂.urm : = p∂.urm ; p∂.urm : =q ;
```

2.3. Structura unui program PASCAL. Blocul

Unitatea de bază a programului este blocul. El cuprinde, pe de o parte, descrierea datelor şi pe de altă parte, acţiunile executate asupra acestora. Descrierea datelor se face prin declaraţii de constante, tipuri şi variabile. Acţiunile sînt cuprinse în corpul de instrucţiuni al blocului. Un bloc poate să conţină la rîndul lui descrierea altor unităţi de program reprezentate de asemenea prin blocuri. Acestea sînt procedurile şi funcţiile (rutinele limbajului). Prin urmare un program PASCAL prezintă o structură ierarhizată de blocuri suprapuse.

Această structură poate fi urmărită şi în programul 1 (fig. 3). Întregul program este un bloc cu partea de declaraţii cuprinsă între liniile 2—17 şi corpul de instrucţiuni între 18—22. Acest bloc conţine blocul corespunzător procedurii „rezolvă" (declaraţii 4—9, corp de instrucţiuni 10—17). În cadrul procedurii se declară încă două blocuri şi anume funcţiile „y" şi „yd", a căror parte de declaraţii este vidă. Executarea programului începe cu corpul de instrucţiuni al blocului exterior (programul principal) — linia 18.

```
 1  PROGRAM NEWTON;
 2     VAR R,START:REAL;
 3     PROCEDURE REZOLVA(VAR RAD:REAL;XI:REAL);
 4        CONST EPS=0.0001;
 5        VAR XO,XD:REAL;
 6        FUNCTION Y(X:REAL):REAL;
 7            BEGIN Y:=COS(X)-X END;
 8        FUNCTION YD(X:REAL):REAL;
 9            BEGIN YD:=-SIN(X)-1 END;
10     BEGIN
11        X1:=XI;
12        REPEAT
13            XO:=X1;
14            X1:=XO-Y(XO)/YD(XO)
15        UNTIL (ABS(X1-XO)<EPS);
16        RAD:=X1
17     END;
18  BEGIN
19     READ(START);
20     REZOLVA(R,START);
21     WRITE('SOLUTIA=',R)
22  END.
```

Fig. 3. (Programul 1). Determinarea rădăcinii ecuaţiei cos(x)=x prin metoda Newton.

Programul calculează rădăcina ecuaţiei cos (x)—x =0, cu aproximaţia „eps" =0.0001, pornind de la valoarea iniţială „start", care se citeşte.

Pentru determinarea aproximativă a rădăcinii se foloseşte metoda lui Newton. Pe baza formulei tg α =y (x0)/(x0—x1), unde tg α este valoarea derivatei

funcției în x0, se calculează valori succesive x1 în funcție de valorile anterioare x0. Calculul se încheie cînd aproximațiile succesive sînt suficient de apropiate.

Procedurile și funcțiile reprezintă blocuri cărora li se asociază la declarare un nume și o listă de parametri formali. La declararea unei funcții se preoizează în plus și tipul rezultatului. Procedurile și funcțiile sînt modulele din care se construiește în mod ierarhizat un program PASCAL.

Activarea unei proceduri se face prin apelarea ei în cadrul unei instrucțiuni de apel (linia 20). O funcție se activează la evaluarea unei expresii în care apare numele ei (linia 14). În ambele cazuri se va executa corpul de instrucțiuni al blocului corespunzător rutinei. Argumentele (parametrii actuali) transmiși la apel trebuie să corespundă ca număr și tip cu parametrii formali specificați la declararea rutinei.

2.4. Tipuri sistem. Modularizarea unui program PASCAL CONCURENT

S-a arătat anterior că prin noțiunea de tip se realizează asocierea între o mulțime de valori și un set de operații permise între valorile respective. În acest sens, tipurilor standard le sînt asociate anumite operații elementare definite prin limbaj. Descrierea operațiilor între structuri rămîne în sarcina programatorului și se realizează în general prin proceduri și funcții adecvate. Prin urmare s-a ivit necesitatea de a introduce construcții de limbaj prin care să se permită asocierea în mod explicit a unor structuri de date cu operațiile aplicabile lor. Se asigură astfel legătura date-acțiuni, în sensul că unor structuri li se pot aplica doar anumite operații bine determinate.

Pornind de la aceste considerente, în limbajul PASCAL CONCURENT s-a extins noțiunea de tip prin introducerea tipurilor sistem *proces*, *monitor* și *class*. O definiție de tip sistem are sintaxa unui bloc și conține prin urmare descrierea unor date (declarații de constante, tipuri, variabile) și a acțiunilor prin care se operează asupra acestora (declarații de proceduri și funcții, corp de instrucțiuni).

În programul 2 (fig. 4) se definesc tipurile clasă „ie-terminal" (liniile 22—75) „ie-banda" (77—94), „ie-consola" (96—127), tipul monitor „acces-banda" (129—151) și tipurile proces „central" (153—183) și „operator" (185—207).

Asupra datelor descrise în interiorul unui tip sistem se poate acționa doar prin intermediul acțiunilor descrise în aceste tipuri. În cazul în care accesul la datele tipului sistem este permis din afara acestuia, prelucrările din exterior se execută apelînd rutinele descrise în tipul sistem și declarate ca externe (declarația conținînd cuvîntul cheie *entry*).

Modulele din care este compus un program PASCAL CONCURENT sînt elemente de tip *process*, *monitor*, *class*. Programul concurent se descrie ca o interconexiune a unor variabile de aceste tipuri sistem.

```
 1  CONST NRPROCESE=3;
 2  CONST EM='(:25:)';CRR='(:13:)' ;LF='(:21:)';NL='(:21:)';TRM='(:11:)';
 3  TYPE OPERATIEIE=(CITIRE,SCRIERE,CONTROL,EDI_C,RASP_C,C_C,REBOBINARE,
 4                 STERGERE,SALT_BLOC,SALT_FISIER,SCR_EOF,POZITIONARE,
 5                 CIT_POZITIE,ARR_ATT_BRK,ATT_BRK,DEZACTIV);
 6  TYPE PARAMIE=RECORD
 7                 OPERATIE:OPERATIEIE;
 8                 STARE,LUNGIME:INTEGER;
 9                 VS:ARRAY(.1..6.) OF CHAR;
10                 DV:ARRAY(.1..4.) OF CHAR;
11                 ABORT:BOOLEAN
12                  END;
13  TYPE APARATIE=(LECTOR,PERF_CART,IMPRIMANTA,CONSOLA,BANDA,DISC,TERMINAL);
14  CONST LUNGLIN=80;
15  TYPE LINIE = ARRAY(.1..LUNGLIN.) OF CHAR;
16       DATA=RECORD AN,LUNA,ZI:INTEGER END;
17  TYPE DOCUMENT=RECORD
18                 C1,C2,C3:LINIE
19              END;
20  CONST LGART=48;
21  [##################################]
22  TYPE IE_TERMINAL=CLASS(NRAPARAT:CHAR);
23  [##################################]
24  VAR PARAMI,PARAME,PARAMBRK:PARAMIE;
25       CONV:CONVERSII; SPECIALE:SET OF CHAR;
26       SPECIALE:SET OF CHAR;
27  PROCEDURE SCRMESAJ1(LIN:LINIE);     [ PROCEDURA INTERNA ]
28  VAR ZONA:LINIE; I:INTEGER; SFR:BOOLEAN;
29  BEGIN
30       SFR:=FALSE; I:=0;
31       WHILE NOT SFR DO BEGIN
32            I:=I+1; ZONA(.I.):=LIN(.I.);
33            SFR:=(I=LUNGLIN) OR (LIN(.I.)='(:0:)');
34       END;  [S-A DETERMINAT LUNGIMEA MESAJULUI CARE SE TERMINA CU (:0:)]
35       PARAME.LUNGIME:=I;
36       IO(ZONA,PARAME,TERMINAL);
37  END;
38  PROCEDURE ENTRY CITMESAJ(VAR LIN:LINIE);
39  VAR ZONA:LINIE; CONTOR:INTEGER; SFR:BOOLEAN;
40  BEGIN
41       IO(ZONA,PARAMI,TERMINAL);
42       IF PARAMI.STARE<>0 THEN BEGIN        [ TRATARE EROARE ]
43            CONTOR:=1;
44            REPEAT
45                 SCRMESAJ1('(:13:)(:21:)REPETATI MESAJUL(:13:)(:21:)(:0:)');
46               IO(ZONA,PARAMI,TERMINAL);
47               CONTOR:=CONTOR+1;
48               IF CONTOR=10 THEN IO(ZONA,PARAMBRK,TERMINAL);
49            UNTIL PARAMI.STARE=0 ;          \ [ SF. TRATARE EROARE ]
50       END;
51       SFR:=FALSE; CONTOR:=0;
52       WHILE NOT SFR DO BEGIN
53            CONTOR:=CONTOR+1;
54            LIN(.CONTOR.):=ZONA(.CONTOR.);
55            SFR:=(CONTOR=LUNGLIN) OR (LIN(.CONTOR.) IN SPECIALE) ;
56       END;
57       IF CONTOR<LUNGLIN THEN FOR CONTOR:=CONTOR TO LUNGLIN DO
58                                LIN(.CONTOR.):=' ';
59  END;
60  PROCEDURE ENTRY ASTBRK;
61  BEGIN IO(PARAMBRK,PARAMBRK,TERMINAL) END;
62  PROCEDURE ENTRY SCRMESAJ(LIN:LINIE);
63  BEGIN SCRMESAJ1(LIN)  END;
64  BEGIN
65       SPECIALE:=(.EM,CRR,LF,NL,TRM.);
66       WITH PARAMI DO BEGIN
67            OPERATIE:=CITIRE;DV:='TT  ';DV(.3.):=NRAPARAT;ABORT:=FALSE;
68            LUNGIME:=LUNGLIN;
69       END;
70       WITH PARAME DO BEGIN
71            OPERATIE:=SCRIERE;DV:='TT  ';DV(.3.):=NRAPARAT;ABORT:=FALSE;
72       END;
73       WITH PARAMBRK DO BEGIN
74            OPERATIE:=ATT_BRK;DV:='TT  ';DV(.3.):=NRAPARAT;ABORT:=FALSE;
75       END;
76  END;
77  [####################]
78  TYPE IE_BANDA=CLASS
79  [####################]
80  VAR PARAM,SFPARAM:PARAMIE;
81  PROCEDURE ENTRY SCRBLOC(VAR ART:DOCUMENT);
82  BEGIN
83       PARAM.STARE:=1; IO(ART,PARAM,BANDA);
84  END;
85  PROCEDURE ENTRY SCREOF;
86  BEGIN IO(SFPARAM,SFPARAM,BANDA) END;
87  BEGIN
88       WITH PARAM DO BEGIN
89            OPERATIE:=SCRIERE; LUNGIME:=LGART;DV:='MT1 '; ABORT:=TRUE;
90       END;
91       WITH SFPARAM DO BEGIN
```

```
 92             OPERATIE:=SCR_EOF;DV:='MT1 ';ABORT:=TRUE;
 93         END;
 94     END;
 95     TYPE MESAJ=ARRAY(.0..LUNGLIN.) OF CHAR;
 96     [####################]
 97     TYPE IE_CONSOLA=CLASS;
 98     [####################]
 99     VAR PARAMI,PARAME,PARAMC:PARAMIE; SPECIALE:SET OF CHAR;
100     PROCEDURE ENTRY SCRMESAJ(LIN:LINIE);
101     VAR MES:MESAJ;I:INTEGER; SFR:BOOLEAN;
102     BEGIN
103         SFR:=FALSE; I:=0;
104         WHILE NOT SFR DO BEGIN
105             I:=I+1; MES(.I.):=LIN(.I.);
106             SFR:=(I=LUNGLIN) OR (LIN(.I.)='(:0:)');
107         END;
108         MES(.0.):=CHR(I); IO(MES,PARAME,CONSOLA);      [LG MESAJ IN MES(.0.)]
109     END;
110     PROCEDURE ENTRY CITMESAJ(VAR LIN:LINIE);
111     VAR MES:MESAJ; I:INTEGER; SFR:BOOLEAN;
112     BEGIN
113         MES(.0.):=CHR(LUNGLIN); IO(MES,PARAMI,CONSOLA);
114         SFR:=FALSE; I:=0;
115         WHILE NOT SFR DO BEGIN
116             I:=I+1; LIN(.I.):=MES(.I.);
117             SFR:=(I=LUNGLIN) OR (LIN(.I.) IN SPECIALE);
118         END;
119         IF I<LUNGLIN THEN FOR I:=I TO LUNGLIN DO LIN(.I.):=' ';
120     END;
121     PROCEDURE ENTRY INTERVENTIE;
122     BEGIN IO(PARAMC,PARAMC,CONSOLA) END;
123     BEGIN
124         SPECIALE:=(.EM,CRR,LF,NL,TRM.);
125         PARAMI.OPERATIE:=CITIRE;PARAMI.ABORT:=TRUE;PARAMI.DV:='TW  ';
126         PARAME.OPERATIE:=SCRIERE; PARAME.ABORT:=TRUE;PARAME.DV:='TW  ';
127         PARAMC.OPERATIE:=CONTROL; PARAMC.DV:='TW  ';
128     END;
129     [##########################]
130     TYPE ACCES_BANDA=MONITOR;
131     [##########################]
132     VAR B:IE_BANDA; I:INTEGER;
133         LOCATII:ARRAY(.1..NRPROCESE.) OF QUEUE;
134         ACTIV:ARRAY(.1..NRPROCESE.) OF BOOLEAN;
135     PROCEDURE ENTRY SCRARTICOL(VAR ART:DOCUMENT;NRP:INTEGER);
136     BEGIN
137         IF NOT ACTIV(.NRP.) THEN DELAY(LOCATII(.NRP.));
138         B.SCRBLOC(ART);
139     END;
140     PROCEDURE ENTRY OPRESTE(NRP:INTEGER);
141     BEGIN ACTIV(.NRP.):=FALSE END;
142     PROCEDURE ENTRY PORNESTE(NRP:INTEGER);
143     BEGIN ACTIV(.NRP.):=TRUE; CONTINUE(LOCATII(.NRP.)); END;
144     PROCEDURE ENTRY INCHIDE;
145     BEGIN
146         FOR I:=1 TO NRPROCESE DO ACTIV(.I.):=FALSE;
147         B.SCREOF;
148     END;
149     BEGIN
150         FOR I:=1 TO NRPROCESE DO ACTIV(.I.):=FALSE;
151         INIT B;
152     END;
153     [######################################]
154     TYPE CENTRAL=PROCESS(ACCES:ACCES_BANDA);
155     [######################################]
156     VAR C:IE_CONSOLA; LIN:LINIE;
157     PROCEDURE CITNRPROC(VAR NRP:INTEGER);
158     VAR LIN:LINIE;
159     BEGIN
160         REPEAT
161             C.SCRMESAJ('DATI NUMARUL PROCESULUI:(:0:)');
162             C.CITMESAJ(LIN);    NRP:=ORD(LIN(.1.))-ORD('0');
163         UNTIL (NRP>=1) AND (NRP<=NRPROCESE);
164     END;
165     PROCEDURE LANSARE;
166     VAR NRP:INTEGER;
167     BEGIN CITNRPROC(NRP); ACCES.PORNESTE(NRP); END;
168     PROCEDURE  OPRIRE;
169     VAR NRP:INTEGER;
170     BEGIN CITNRPROC(NRP); ACCES.OPRESTE(NRP); END;
171     PROCEDURE INCHIDE;
172     BEGIN ACCES.INCHIDE END;
173     BEGIN
174         INIT C;
175         CYCLE
176             C.INTERVENTIE;
177             C.SCRMESAJ('COMANDA:(:0:)');
178             C.CITMESAJ(LIN);
179             IF LIN(.1.)='L' THEN LANSARE ELSE
180             IF LIN(.1.)='O' THEN OPRIRE ELSE
181             IF LIN(.1.)='I' THEN INCHIDE ELSE
182             C.SCRMESAJ('EROARE(:0:)');
183         END;
184     END;
185     [##########################################]
```

ARTICOLUL CONTINUĂ LA PAGINA 354

„O punte între știință și tehnologie"

(Congresul mondial trienal I.F.A.C., Budapesta, 1984)

SISTEM DE CONDUCERE CU MICROCALCULATOARE, TOLERANT LA DEFECTE, AL MAȘINII DE ÎNCĂRCARE-DESCĂRCARE A COMBUSTIBILULUI PENTRU CENTRALE NUCLEARE

J. Borka și S. Keresztely

Instit. de Automatizare și Calculatoare, Academia Ungară de Științe (R.P.U.)

Rezumat :

Sistemul de conducere (comandă) elaborat pentru mașina de încărcare descărcare a combustibilului pentru centrale nucleare cuprinde 7 microcalculatoare. Structurile hardware și software redundante asigură o mare fiabilitate și disponibilitate. Senzorii sînt dublați și fiecare set este conectat la propriul său microcalculator, care asigură evaluarea măsurărilor.

Programul de conducere care coordonează 7 acționări electrice este derulat pe două microcalculatoare identice și are acces la ambele seturi de rezultate de măsurare. Există două console operator. Cel de-al șaptelea microcalculator generează un plan simulat al zonei de lucru în jurul poziției reale, iar imaginea este transmisă monitoarelor TV-color de pe pupitrele de comandă.

Fiabilitatea sistemului : la defectarea oricăreia dintre părțile sistemului se produce un mesaj de eroare și nici o acțiune. *Disponibiliatea sistemului :* în caz de avarie, unul din fiecare pereche de subsisteme identice trebuie să fie operațional. În acest regim de avarie se pierde fiabilitatea necondiționată.

Sistemul de antrenare conține acționări de c.c. în patru cadrane, cu comanda curentului de circulație.

Cuvinte cheie : Sisteme redundante (tolerante la defecte) ; fiabilitate ; disponibilitate ; conducere comandă prin microcalculatoare ; acționari de c.c. în patru cadrane.

Introducere : Mașina de încărcare-descărcare a combustibilului (MID) dintr-o centrală nucleară este supusă unor standarde de securitate ridicată tipice industriei centralelor nucleare. Condițiile principale pentru sistemul de comandă al MID sînt :

— disponibilitate foarte mare,
— fiabilitate ridicată,
— posibilitatea comenzii la distanță,
— blocarea comenzilor eronate.

Disponibilitate foarte mare

MID este o macara mare, cu un braț manipulator de precizie. MID trebuie să fie în stare de funcționare atîta vreme cît părțile mecanice și motoarele electrice rezistă la acțiunea mediului ambiant.

Sistemul de comandă trebuie să se comporte în consecinţă.

Nivelul de disponibilitate cerut presupune că, în cazul unei singure erori în oricare parte a sistemului de comandă, disponibilitatea se menţine, cu o siguranţă în funcţionare mai scăzută.

Fiabilitate ridicată

Se cere ca o singură eroare în oricare parte a sistemului de comandă, însoţită sau nu de o comandă eronată sau corectă a operatorului, să nu ducă la o acţiune greşită.

Posibilitatea comenzii la distanţă

Ţinînd seama de mediul periculos în care funcţionează MID, este necesară şi o consolă de comandă la distanţă. Pentru informarea completă a operatorului, la acest pupitru trebuie să existe toate afişajele, un ecran TV în circuit închis etc.

Proiectanţii consideră că această consolă pentru comandă la distanţă trebuie utilizată şi pentru operaţiunile normale de comandă a maşinii.

Blocarea comenzilor greşite

Braţul MID se mişcă într-un spaţiu multidimensional, cu mai multe limitări ale mişcării. Depăşirea acestor limitări poate duce la deteriorarea barei de combustibil, a MID sau a ambelor. Sistemul de comandă verifică comenzile operatorului, în funcţie de aceste limitări (geometrice şi altele, de ex. greutăţi sau secvenţe operaţionale) şi blochează execuţia în cazul nerespectării lor.

Structura sistemului de comandă

Sistemul de comandă şi acţionările electrice sînt amplasate în camera de comandă la distanţă, unde condiţiile de mediu sînt mai puţin severe. Senzorii de poziţie, comutatoarele de contact şi tensometrele, motoarele şi frînele montate pe MID sînt de tipul „regim greu".

Sistemul de comandă conţine subsisteme de patru tipuri diferite iar cele trei tipuri de subsisteme vitale sînt dublate, după cum se arată în fig. 1. Subsistemele dublate execută aceleaşi sarcini şi trebuie să producă ieşiri identice.

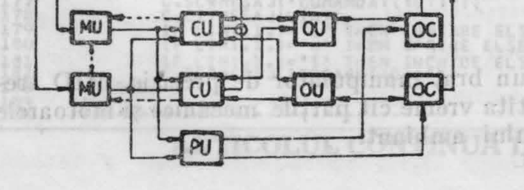


Fig. 1. Structura sistemului de comandă.

Blocul de măsurare (MU)

Ambele blocuri de măsurare își au propriul lor set de senzori pe MID. MU convertește semnalele de fază ale transformatoarelor de comandă a sesizării poziției în coordonate, face verificările logice ale acestor valori și stărilor comutatoarelor de contact și compilează un mesaj de ieșire la fiecare 20 ms.

Acest mesaj conține informațiile de stare referitoare la microcalculatorul blocului de măsurare, la sistemul de măsurare, și cum noile rezultate ale măsurării.

Mesajul este însoțit de o verificare a redundanței ciclice pe 16 biți și este transmis la alte unități fără a aștepta o confirmare. În cazul unei erori de comunicație detectate la capătul de recepție, mesajul trebuie — pur și simplu — anulat.

Informațiile pierdute, se repetă în cursul cîtorva cicluri de măsurare. Traseele de informații, reprezentate în fig. 1 prin linii întrerupte, nu se utilizează în timpul funcționării MID, ele servesc pentru testare.

Blocul de comandă (CU)

CU primește informații de la blocul de măsurare, comunică cu operatorul (primind comenzi și trimițînd răspunsuri) și execută algoritmul de comandă.

Funcțiuni de supraveghere. CU primește comenzi de la operator cu privire la starea de funcționare a sistemului, ține evidența informațiilor de stare a blocului de măsurare și a celuilalt bloc de comandă.

Din acestea se obține starea prezentă a sistemului și se transmite la consola operatorului. Dacă cerințele sistemului nu sînt corespunzătoare se generează un mesaj de eroare.

Funcțiuni de evaluare a comenzii. În regim manual, operatorul inițializează o mișcare, dînd o comandă corespunzătoare mașinii, în timp ce, în regim automat, comenzile sînt generate de către interpretorul de comenzi al blocului de comandă. În ambele cazuri comenzile se testează față de limitele geometrice și de alte limite. Dacă testul reușește, comanda se asociază cu parametrii limită de curent și viteză și cu poziția limită în care trebuie să se oprească acționarea.

După suplimentarea cu aceste informații, comanda este gata de execuție.

Funcțiuni de comandă propriu-zise. În blocul de comandă, acest task compară între ele mesajele de la blocul de măsurare cu informațiile anterioare și actualizează setul de date de măsurare. Dacă setul de date de măsurare este corect, se pot transmite comenzi la acționările electrice. La apropierea de poziția limită, viteza mișcării se reduce pas cu pas și, în ultima fază, funcția de evaluare a comenzii se reinițializează. Progresul realizat în ciclul de lucru poate să aibă ca rezultat noi poziții limită sau alți parametri. Inițializarea repetată a evaluării comenzii are ca rezultat execuția unei secvențe de comenzi cu depășire maximă, dirijate de regulile de limitare.

Pupitrul operatorului (OU)

Există în sistem două pupitre ale operatorului, însă numai unul dintre ele poate fi activ la un moment dat. Cel pasiv — dacă este alimentat — afişează informaţiile şi nu face nimic altceva.

Funcţia principală a OU constă în comprimarea/expandarea informaţiilor între blocul de comandă, afişajul operatorului şi tastatură. Pentru transmiterea informaţiilor şi pentru asigurarea eficienţei prelucrării, mesajele la blocul de comandă sînt codificate în formă compactă, în timp ce operatorul are la dispoziţie textul în clar.

Consola operatorului conţine afişajul şi tastatura, elementele de comandă manuală conectate direct la acţionările electrice, TV-ul în circuit închis şi monitorul color conectat la blocul video.

Blocul video (PU)

Blocul video nu face parte din sistemul de comandă, ci este un ajutor vizual pentru operator. Primeşte rezultatele măsurătorii de la blocul de măsurare şi generează un plan simulat al zonei din jurul braţului manipulator al MID.

Locaţiile unde pot fi găsite sau aşezate barele de combustibil sînt arătate pe monitorul color prin cercuri colorate.

Imaginea constă din 256×384 puncte şi poate fi deplasată prin hardware cu o rezoluţie de un punct echivalent cu cca 5 mm.

Funcţionarea sistemului

În sistem funcţionează două blocuri de măsurare, două blocuri de comandă şi pupitrul operatorului. Cele două blocuri de măsurare primesc datele de măsurare corecte şi rezultatele calculate sînt transmise ambelor blocuri de comandă.

Blocurile de comandă primesc informaţiile de la ambele blocuri de măsurare şi le compară. În urma comparării, în ambele blocuri de comandă, în cazul echivalenţei informaţiilor, algoritmul de comandă este activat.

Semnalele de comandă calculate sînt transmise la acţionările electrice prin ambele blocuri de comandă prin optoizolatoare conectate în serie. Semnalul de ieşire combinat este reţinut de blocurile de comandă şi nici unul dintre blocurile de comandă nu detectează limitarea de către celălalt, adică ieşirile sînt aceleaşi.

Comenzile de la pupitrul operatorului activ sînt primite şi confirmate de ambele blocuri de comandă.

Mesajele către pupitrul operatorului sînt transmise numai printr-un singur bloc de comandă. Pupitrul operatorului nu este dublat. Nu există nici un motiv pentru a face acest lucru : sistemul este asigurat împotriva comenzilor

greșite din partea operatorului și cel mai rău lucru pe care-l poate face o unitate operator defectă este să producă un asemenea mesaj de ieșirea.

Se poate vedea cu ușurință : în acest sistem, orice eroare însoțită de o comandă greșită din partea operatorului nu poate niciodată să inițieze o acțiune greșită. Cel puțin unul dintre blocurile de comandă este corect și primește cel puțin un set de rezultate corecte ale măsurării. Înseamnă că acest bloc de comandă nu acționează asupra unui set greșit de rezultate ale măsurării și blochează astfel comenzile eronate ale operatorului (sau OU generează comenzi false) și oprește orice acțiune greșită a celuilalt CU de comandă.

Sistemul este asigurat și împotriva dublelor greșeli, cu excepția cazului următor : aceeași eroare este generată în ambele subsisteme (duble) în intervalul timpului de reacție al sistemului.

Sistemul reacționează la erori cu toleranța necesară : încearcă să compenseze erorile de comunicație dintre subsisteme. Durata unui ciclu al programului blocului de măsurare și comunicației este de 20 ms ; se tolerează unul sau două cicluri, în funcție de natura datelor ce lipsesc. Timpul de comunicație al OU și al ciclului de evaluare a comenzii de cca 100 ms și se tolerează două mesaje lipsă sau incorecte.

Semnalele de ieșire către acționările electrice pot să difere timp de cca 200 ms. Limita de curent și semnalele de viteză sînt codificate : o diferență de 1 bit în valoarea codificată nu este tratată ca o nepotrivire. În acest mod, se evită reacția de panică față de erori temporare nesemnificative.

Pentru detectarea prematură a erorilor (în părțile nefolosite ale sistemului) fiecare microcalculator execută un program de autotestare la nivelul cel mai mic de prioritate, iar dezvoltarea acestui program este verificată la nivelul superior. Acest program de autotestare citește memoria de la început la sfîrșit și în caz de eroare la verificarea de paritate oprește microcalculatorul. Blocurile de măsurare trimit informațiile lor de stare la unitățile de comandă.

Cele 2 blocuri de comandă schimbă informații de stare regulat prin intermediul OU activ și verifică ciclic interfața cu acționările electrice pentru evitarea scurtcircuitelor.

Funcționarea cu subsisteme defecte

Imediat ce se detectează un defect se trimite un mesaj de eroare la consola operatorului și mașina de încărcare-descărcare a combustibilului se oprește. Personalul trebuie să repare defectul (să înlocuiască placa cu circuite imprimate cu una de rezervă). Dacă, pentru anumite motive, înlocuirea nu este posibilă, sistemul este totuși capabil să funcționeze la un nivel mai scăzut de siguranță. Pentru inițializarea acestor stări de funcționare este necesară aplicarea unei comenzi.

Erori de măsurare

Dacă unul dintre blocurile de măsurare furnizează o informație greșită blocurilor de comandă, aceasta poate fi exclusă din sistemul de comandă prin alegerea stării de funcționare a unui singur bloc de măsurare. Dezavantajul :

orice eroare nouă de măsurare care nu este detectată în blocul de măsurare poate fi o sursă de eroare nedetectată.

Dacă într-unul din blocurile de măsurare una sau mai multe măsurători sînt greșite este posibil să se aleagă blocuri de măsurare combinate. În acestă stare de funcționare, numai măsurătoarea detectată (și confirmată de operator) este unică, toate celelalte sînt verificate din punct de vedere al egalității în cele două blocuri de comandă.

Dezavantajul este mai mic decît în cazul unui singur bloc de măsurare : numai perechea măsurătorii care lipsește poate cauza o eroare nedetectată.

Erorile dispozitivului de comandă

Dacă unul dintre blocurile de comandă nu funcționează adecvat, operatorul o poate exclude din sistemul de comandă alegînd starea de funcționare a unui singur bloc de comandă. Dacă microcalculatorul exclus mai este încă în funcționare, el înțelege această comandă și trece în stare de rezervă ; își scurtcircuitează ieșirile la acționările electrice și încetează să se interfereze cu sistemul.

Dacă microcalculatorul este incapabil să execute această comandă de rezervă, trebuie izolat din sistem. Aceasta se poate face prin rearanjarea cablurilor conectînd direct acționările electrice cu blocul de control de funcționare.

Dezavantaj : erorile din blocul de comandă nu sînt detectate de o unitate independentă.

Defectarea consolei operatorului

În acest caz sistemul trebuie să fie deconectat și conectat la cealaltă consolă.

Consola la care a fost conectat sistemul este automat consolă activă și sistemul nu solicită starea de funcționare a celeilalte console. Se asigură o fiabilitate completă.

Testarea sistemului

Testarea de hardware și software pînă la nivel de subsistem individual se face în mod obișnuit. Testarea nivelului sistemului a pus o problemă serioasă, pentru că sistemul de comandă nu a putut fi conectat la mașina de încărcare-descărcare a combustibilului pentru test, dar oricum, fără o platformă de testare la fel de mare ca platforma dintr-o centrală electrică nu ar fi de mare ajutor.

S-a hotărît să se simuleze mașina de încărcare-descărcare a combustibilului. Un program complet de simulare pentru mașina de încărcare-descărcare a combustibilului și a mediului înconjurător ar fi o problemă grea cu multe posibilități de greșeală.

Considerînd că nu este necesar să se simuleze situații care au loc numai atunci cînd sistemul de comandă nu funcționează (ex. lovirea de obstacole) problema a devenit foarte simplă.

Funcţiile care trebuie realizate :

— generarea valorilor măsurate în concordanţă cu comenzile trimise la acţionările electrice :

— oprirea acţionărilor într-o anumită poziţie şi indicarea „limitei de curent atinsă" ;

— schimbarea rezultatelor măsurării greutăţii datorită mişcărilor simulate.

Aceste funcţii simple sînt realizate în cadrul blocurilor de comandă şi a blocurilor de măsurare cu sarcini independente. Pentru a obţine o soluţie corectă, task-urile de simulare nu împart informaţiile cu alte programe din microcalculator şi folosesc un canal de date separat pentru comunicare (arătat în fig. 1 prin linia întreruptă).

Înainte de a începe simularea, cablurile către acţionările electrice trebuie înlocuite cu un dispozitiv de testare.

Programul de simulare din blocul de comandă poate fi început prin panoul operator al microcalculatorului. El verifică prezenţa dispozitivului de testare lucrînd în absenţa acţionărilor reale. Programul de simulare citeşte comenzile care sînt ieşirile combinate ale celor două blocuri de comandă. În concordanţă cu aceste comenzi se alege viteza de simulare şi se transmite la unul din blocurile de măsură. Acest bloc de măsură — dacă simularea a fost începută la panoul operator al microcalculatorului — generează rezultatele măsurării şi le trimite direct la blocurile de comandă prin intermediul celuilalt bloc de măsură. În timpul simulării, totul pare normal la consola operatorului, ca şi cum maşina de încărcare-descărcare a combustibilului ar funcţiona.

Răspunsurile dinamice sînt oarecum diferite, dar toate funcţiile de securitate şi limitările de mişcare există, împreună cu mesajele de eroare.

Monitorul color indicînd zona din jurul poziţiei braţului maşinii de încărcare-descărcare a combustibilului, este de asemenea în funcţiune.

Acest software de simulare a devenit o parte din ansamblul final de software. Poate fi folosit pentru a demonstra funcţiile sistemului şi pentru verificare. Un alt domeniu de utilizare este pregătirea operatorilor.

Realizare :

Sistemul de comandă constă din şapte subsisteme. Fiecare subsistem este construit în jurul unui microcalculator de acelaşi tip şi sînt adăugate circuite de interfaţă pentru a specializa subsistemul. Microcalculatoarele includ un microprocesor Z80, memorie EPROM de 32 Ko şi CMOS RAM de 8 Ko cu control de paritate, pentru porţi seriale de I/E şi un panou de operare.

Unităţile sînt montate în sertar folosind plăci cu circuite imprimate Eurocard.

Subsistemele comunică între ele prin porţile seriale de I/E ale microcalculatoarelor, folosind protocolul de linie HDLC.

Blocul de măsură este arătat în fig. 2. Cartela de intrare primeşte un semnal de fază de referinţă şi foloseşte un multiplicator de frecvenţă PLL şi contoare de 8 biţi pentru măsurarea semnalului de fază al transformatoarelor de comandă.

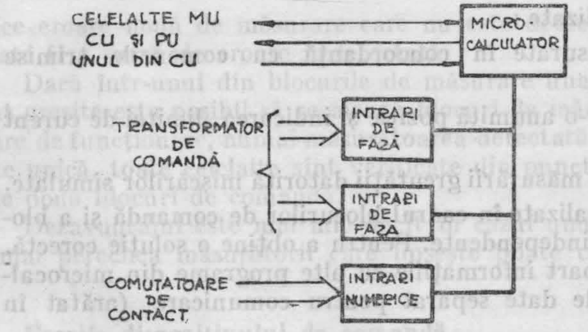


Fig. 2. Blocul de măsură.

Blocul de comandă este arătat în fig. 3. O cartelă de interfață poate comanda două acționări electrice, cele șapte acționări ale mașinii de încărcare-descărcare necesită patru cartele.

Pupitrul operatorului este arătat în fig. 4.

Cartela de comandă a display-ului conține memoria de recirculare și generatorul de caractere.

Blocul video este arătată în fig. 5.

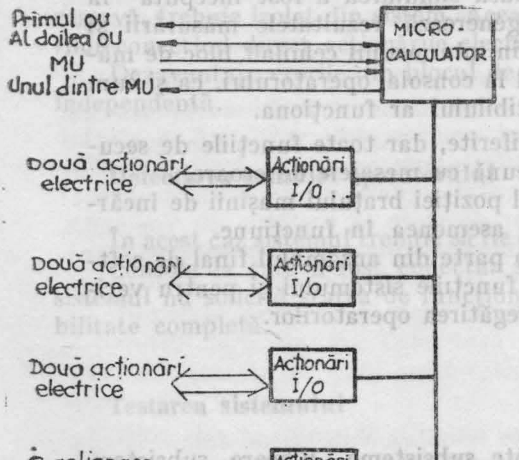


Fig. 3. Blocul de comandă.

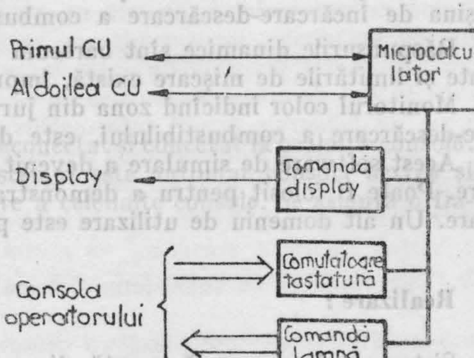


Fig. 4. Pupitrul operatorului.

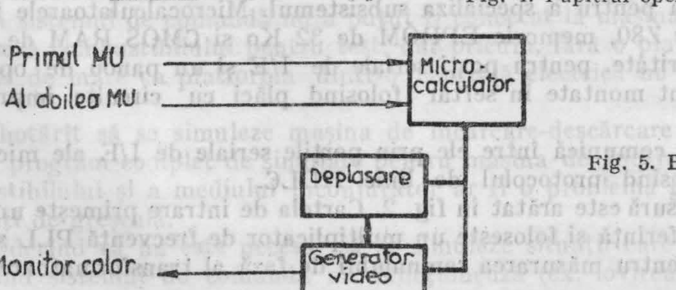


Fig. 5. Blocul video.

Cartele generatorului video conţine memoria de recirculare şi generatorul setului de caractere semigrafice.

Sistemul de acţionare (DS)

Sarcina sistemului de acţionare este de a asigura mişcarea comandată în şapte direcţii a maşinii de încărcare-descărcare a combustibilului atît în regim manual cît şi automat. În regimul manual, operatorul direcţionează mişcarea maşinii de încărcare-descărcare a combustibilului (MID) de la o consolă a operatorului OC la alta, dar sistemul de comandă supraveghează permanent întregul ciclu de lucru şi comenzile operatorului.

În regim automat consola operatorului este folosită direct de operator pentru a direcţiona sistemul de comandă pentru realizarea ciclului de lucru dorit.

Aceasta se face prin darea comenzilor necesare către acţionările individuale.

Sistemul de acţionare cuprinde şase acţionări comandate în c.c. şi o acţionare de c.a. fără comanda vitezei. Aceste acţionări sînt următoarele:

— acţionare de cursă lungă;
— acţionare de traversare;
— acţionare de ridicare, echipată cu şină cu cremalieră;
— acţionare de ridicare, echipată cu cablu;
— acţionare de rotaţie pentru mişcarea de rotaţie a tijei de lucru;
— acţionarea de rotaţie pentru mişcarea de rotaţie a platformei TV;
— acţionarea de ridicare fără comanda vitezei pentru mişcarea verticală a tijei TV.

Schema bloc a unei acţionări individuale poate fi văzută în fig. 6.

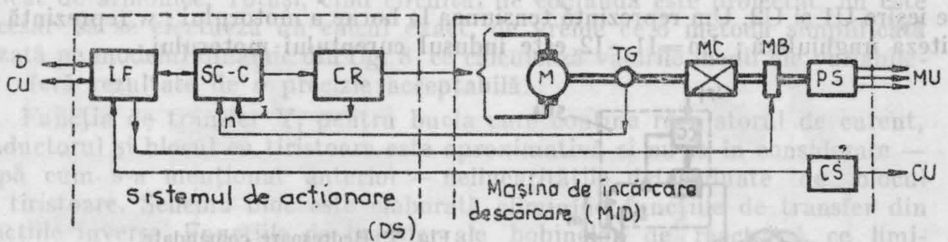


Fig. 6. Schema bloc a sistemului de acţionare.

Unitatea de interfaţă (IF) evaluează semnalele care vin atît de la consola operatorului (OC) şi de la blocul de comandă (CU) şi în plus furnizează informaţiile de stare ale acţionărilor individuale către blocul de comandă şi transmite valoarea curenţilor motoarelor consolei operatorului (OC).

Fiecare ansamblu de acţionare trebuie să fie capabil să acţioneze şi să frîneze în ambele sensuri de rotaţie, respectiv sînt necesare acţionări în patru cadrane.

O serie de echipamente tehnologice antrenate ce necesită acţionări în patru cadrane, din motive de siguranţă impun o trecere prin zero continuă

a curentului. Acesta este motivul pentru care acţionările simple în curent continuu fără curent de circulaţie nu au fost selectate deşi ele posedă proprietăţi preferabile în alte privinţe. Mecanismul de ridicat necesită un cuplu nominal chiar şi la viteza zero a maşinii de încărcare-descărcare a combustibilului, şi pentru a obţine o uniformitate, toate acţionările sînt de acelaşi tip.

În acţionările în curent continuu cu curent de circulaţie, două seturi de redresoare comandate sînt conectate antiparalele prin intermediul bobinelor de reactanţă şi un set de redresoare asigură curentul de circulaţie. În practică există două soluţii diferite pentru comanda de amorsare a acestor acţionări. În cazul uneia dintre acestea, unghiul de amorsare al redresoarelor comandate, variază în funcţie de un raport selectat într-un mod adecvat, iar mărimea curentului de circulaţie într-un regim de lucru discontinuu este limitată doar de inductanţa bobinelor de reactanţă.

Dezavantajele acestei soluţii sînt după cum urmează : curentul de circulaţie depinde de unghiul de amorsare ; în regimul de lucru continuu componenta sa de c.c. este limitată numai de rezistenţa circuitului de c.c., conţinînd în primul rînd bobinele de reactanţă. În cazul altor soluţii există o dispunere specială de control a curentului de circulaţie.

Structura regulatorului de curent este relativ complicată, deoarece valoarea instantanee a curentului de circulaţie mai depinde şi de sarcină. În plus, nu este simplu de măsurat curentul de circulaţie şi de selectat redresorul, furnizînd numai curentul de circulaţie.

În cazul nostru a fost realizat un dispozitiv special de comandă al curentului de circulaţie şi al motorului, pe baza componentelor simetrice ale curentului. Acţionările de c.c. în patru cadrane constau din două redresoare identice (CR1 şi CR2) cu bune proprietăţi dinamice şi este posibil să se utilizeze traductoare clasice de curent (vezi fig. 7). Semnalele de referinţă de curent Ir1 şi Ir2 prescriu valorile medii ale curenţilor lor de ieşire I1 şi I2 iar dispozitivele lor de comandă forţează aceşti curenţi, independent de tensiunile lor de ieşire U1 şi U2. Um reprezintă tensiunea la borne a motorului ; w reprezintă viteza unghiulară ; Im = I1 − I2 este indusul curentului motorului.

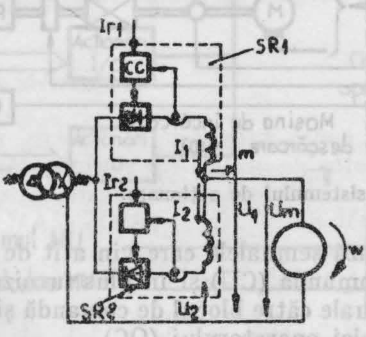


Fig. 7. Redresoare comandate.

Cercetările teoretice au arătat (1) că semnalul de referinţă Irp al curentului în secvenţă pozitivă este proporţional cu suma semnalului de referinţă Irc al curentului de circulaţie şi valoarea absolută a curentului Im al motorului în timp ce semnalul se referinţă Irn al curentului în secvenţă negativă este proporţional cu semnalul de ieşire al regulatorului de viteză. Semnalele de

referință ale curentului celor două redresoare controlate pot fi compuse din semnale de referință ale curentului în secvență pozitivă și negativă prin intermediul unei transformări matriciale corespunzătoare, vezi fig. 8.

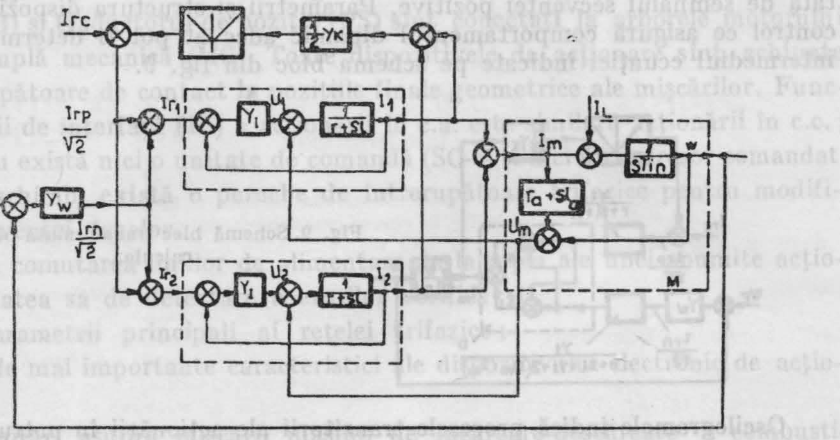


Fig. 8. Modelul dinamic al redresoarelor comandate.

Figura 8 prezintă modelul dinamic potrivit pentru calcularea aproximativă a proceselor tranzitorii din montajul nostru de acționare în curent de circulație.

Figura 8 prezintă influența reciprocă a redresoarelor comandate ce lucrează asupra aceleiași sarcini. În realitate sistemul este nelinear datorită convertizorului cu tiristoare acesta avînd de exemplu un timp mort și un conținut ridicat de armonice. Totuși, cînd circuitul de comandă este proiectat, nu este necesar să se efectueză un calcul exact, de vreme ce o metodă simplificată bazată pe modelul dinamic din fig. 8 ce calculează valorile medii ale variabilelor oferă rezultate de o precizie acceptabilă.

Funcția de transfer Y_I pentru bucla care conține regulatorul de curent, traductorul și blocul cu tiristoare este aproximativă și nu ia în considerare — după cum s-a menționat anterior — nelinearitățile determinate de blocul cu tiristoare. Schema bloc este elaborată eliminînd funcțiile de transfer din reacțiile inverse. Funcțiile de transfer ale bobinelor de reactanță ce limitează curentul de circulație sînt redate prin raportul: $\frac{1}{r+s.1}$. Litera M simbolizează motorul; unde: ra, la, sînt rezistența și respectiv inductanța motorului; Tin reprezintă timpul său nominal de pornire. Efectul de linearizare este exprimat prin funcțiile de transfer Yk; iar I_L simbolizează efectul sarcinii.

După o oarecare modificare a fig. 8 obținem schema bloc transformată din fig. 9, care este utilă pentru a proiecta dispozitivele de control și pentru a analiza limitele de stabilitate și comportamentul dinamic al sistemului de acționare. Porțiunea reprezentată de secvența pozitivă și negativă a circuitului de comandă a fost separată în fig. 9.

Blocul comun D din fig. 9 reprezintă un factor de transfer de ordinul 2 în bucla secvenţei negative.

Din cele de mai sus rezultă că secţiunea din schema bloc ce prescrie curentul de circulaţie, nu este o buclă închisă, deşi există o reacţie inversă reprezentată de semnalul secvenţei pozitive. Parametrii şi structura dispozitivelor de control ce asigură comportamentul dinamic adecvat pot fi determinaţi prin intermediul ecuaţiei indicate pe schema bloc din fig. 9.

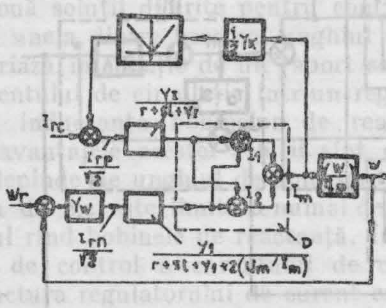


Fig. 9. Schemă bloc transformată pentru calcule.

Oscilogramele indică procesele tranzitorii ale acţionării în patru cadrane, realizate pe baza principiului descris. Modificarea curentului motorului Im şi a vitezei motorului n sînt indicate în fig. 10, în timpul unui proces de inversare a vitezei la o sarcină nominală.

Pentru acelaşi proces, fig. 11 prezintă curentul celor două redresoare Ip şi In. Curentul mai ridicat este totdeauna curentul motorului, iar cel mai scăzut este curentul de circulaţie.

Se poate vedea din oscilogramă că, curentul de circulaţie este practic constant chiar pe perioada unei stări tranzitorii.

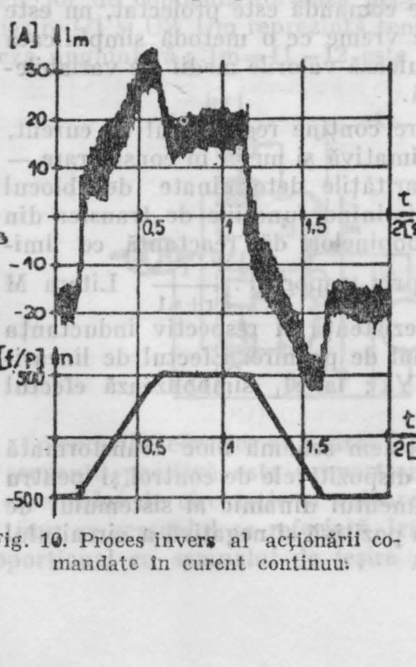


Fig. 10. Proces invers al acţionării comandate în curent continuu.

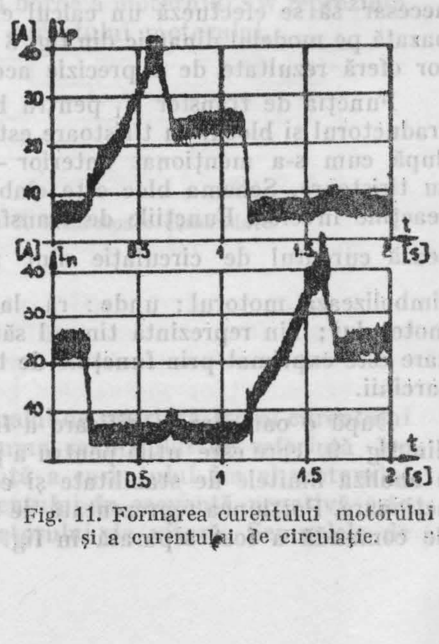


Fig. 11. Formarea curentului motorului şi a curentului de circulaţie.

Sistemul de acţionare în curent continuu cu curent de circulaţie, asigurînd funcţionarea în patru cadrane cu trecere continuă prin zero a curentului, include două dispozitive de comandă (SC-C, viteză şi respectiv curent), vezi fig. 6. Redresorul comandat (CR) furnizează energie servomotorului (M) de c.c. cu magnet permanent, cu tahogenerator (Tg) încastrat. Frîna electromagnetică (MB) şi traductorul de poziţie (PS) sînt conectaţi la arborele motorului printr-o cuplă mecanică (MC). Toate dispozitivele de acţionare sînt echipate cu întrerupătoare de contact la poziţiile finale geometrice ale mişcărilor. Funcţia unităţii de interfaţă (IF) a acţionării în c.a. este similară acţionării în c.c. ; normal, nu există nici o unitate de comandă (SC-c) şi nici un redresor comandat (CR), în schimb, există o pereche de întrerupătoare trifazice pentru modificarea secvenţei fazelor.

După comutarea liniilor de alimentare de la reţea ale unei anumite acţionări, unitatea sa de detectare a erorilor verifică ;

— parametrii principali ai reţelei trifazice ;

— cele mai importante caracteristici ale dispozitivului electronic de acţionare ;

— limitări asupra mişcării maşinii de încărcare-descărcare a combustibilului.

Dacă nu se detectează defecte, atunci mişcarea dorită este permisă şi se va trimite un mesaj către blocul de comandă (CU) cu privire la starea acţionării selectate. După aceasta, operarea acţionării este posibilă atît în regim automat, cît şi manual. În cazul operării de regim automat, comenzile de la blocul de comandă (CU) vor fi verificate de unitatea de interfaţă (IF).

Combinarea incorectă a comenzilor către dispozitivele de comandă ale acţionării (SC-C) va fi blocată de (IF). În cazul unei operări normale (lipsită de greşeli) a dispozitivelor de acţionare, unitatea de interfaţă (IF) furnizează informaţii de stare de la dispozitivele de acţionare la blocul de comandă (CU). Sistemul de comandă al maşinii de încărcare-descărcare a combustibilului, poate permite sau interzice funcţionarea dispozitivelor de acţionare pe baza comparării comenzilor de la blocul de comandă (CU) cu informaţiile de stare de la sistemul de acţionare (DS).

De îndată ce se detectează o eroare în regimul de funcţionare, mişcarea va fi imediat oprită şi se va genera un mesaj de eroare la sistemul de comandă.

Mesajul de eroare va apare pe display-ul consolei operator (OC). În acest caz operatorul sau depanatorul trebuie să se ducă la camera de comandă, pentru a detecta şi înlătura defecţiunea din dulapul de comandă. În cazul apariţiei unei defecţiuni de orice natură şi de orice amploare se va aprinde LED-ul roşu al lămpii de reanclanşare, iar funcţionarea acţionării selectate poate fi reluată numai dacă defecţiunea a fost înlăturată prin apăsarea butonului de reanclanşare.

Concluzii

Sistemul de comandă are o fiabilitate şi disponibilitate ridicată. Costurile moderate se datorează utilizării exclusive a aceluiaşi tip de microcalculator şi faptul că sînt necesare numai două microcalculatoare pentru fiecare subsistem. Acţionările în c.c. în patru cadrane, cu comanda curentului de circulaţie asigură un comportament dinamic de calitate.

BIBLIOGRAFIE

(1) *Brausz I., J. Borka* şi *A. Varga* (1982). **Comanda curentului de circulaţie prin componente simetrice ale acţionăriior în curent continuu în patru cadrane.** ICEM 1982, Budapesta, 141—144.

TENDINȚE ÎN EDUCAȚIA ÎN AUTOMATIZĂRI

Z. Benyó *
Universitatea Tehnică din Budapesta (R. P. Ungară)
Ing. *Marcel Sîrbu*
I.P.A.
Dr. ing. *Adrian Davidoviciu*
I.T.C.I.

I) EDUCAȚIA ÎN R.P. CHINEZĂ, AUSTRIA, S.U.A., JAPONIA, EUROPA DE VEST

Sesiunea 07.1 de la Congresul IFAC a dezbătut prin referatele prezentate în cele ce urmează chiar tendințele în educația în automatizări.

Lista referatelor prezentate a fost :

1. Stabilizarea unui pendul dublu invertor prin regulator analogic ; autori : Z. Feng, Z. Yin, H. Chen (R.P. Chineză) ;
2. Un experiment de laborator controlat de calculator ; autori : C. Wang (R.P. Chineză), R. Henry, R. Cameron. S. Mossaheb (Anglia) ;
3. Educația în reglarea automată în Austria ; autori : P. Kopacek, R. Genser, I. Troch, A. Weinmann (Austria) ;
4. Tendințe educaționale în automatizări în SUA ; autori : A. Desrochers, G. Saridis (SUA) ;
5. Tendințe în educația în automatizări în Japonia ; autori : H. Akashi (Japonia) ;
6. Orizonturi în educația inginerească — o prezentare sistemică a situației în Europa de vest ; autori : G. Battersby, D. Jenkins (Anglia), W. Schaufelberger (Elveția).

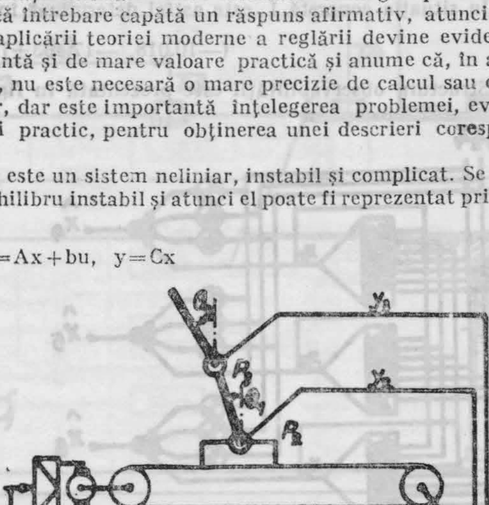
Referatul 07.1—1 prezintă implementarea unui sistem de reglare cu un regulator analogic, atît parțial observabil, cît și linear funcțional observabil. Sistemul prezentat este foarte stabil la perturbații mari și la variații ale parametrilor și este foarte indicat pentru experimentele de educație în teoria modernă a reglării, suscitînd un interes major din partea studenților.

Autorii pornesc de la ideea posibilității utilizării unui calculator analogic pentru rezolvarea problemei și susțin că, dacă această întrebare capătă un răspuns afirmativ, atunci costul regulatorului scade simțitor și scopul aplicării teoriei moderne a reglării devine evident. În completare ei enunță o altă idee interesantă și de mare valoare practică și anume că, în aplicarea practică a teoriei moderne a reglării, nu este necesară o mare precizie de calcul sau cerințe stricte de implementare pentru regulator, dar este importantă înțelegerea problemei, eventual cercetarea și analiza atentă a sistemului practic, pentru obținerea unei descrieri corespunzătoare a regulatorului.

Descriptiv, sistemul pendul (Fig. 1) este un sistem neliniar, instabil și complicat. Se presupune linearizarea în jurul punctului de echilibru instabil și atunci el poate fi reprezentat printr-un model liniar de ordinul șase, astfel :

$$\dot{x}=Ax+bu, \quad y=Cx \tag{1}$$

Fig. 1. Diagrama schematică a pendulului dublu invertor.



* Lucrarea autorului (secțiunea II) mai amplă în original, este restrînsă la informațiile necesare cititorului seriei AMC.

unde :

$$x=(r,\ \theta1,\ \theta2-\theta1,\ \dot{r},\ \dot{\theta}1,\ \dot{\theta}2-\dot{\theta}1)^T$$

$$y=(y1,\ y2,\ y3)^T=(r,\ \theta1,\ \theta2-\theta1)^T$$

$$A=\begin{pmatrix} 0 & I \\ A_{21} & A_{22} \end{pmatrix}=$$

$$=\begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1{,}982 & 0{,}09565 & -9{,}550 & 0{,}00817 & -0{,}036 \\ 0 & 26{,}871 & 13{,}85000 & 21{,}76 & -0{,}1602 & 0{,}13135 \\ 0 & 31{,}19 & 57{,}847 & 25{,}27 & 0{,}35027 & -0{,}4292 \end{pmatrix}$$

$$b=(0,\ b_2)^T=(0,\ 0,\ 0,\ 6.1836,\ -14.096,\ 16.361)^T$$

$$C=(I,\ 0)$$

şi u este tensiunea de intrare a amplificatorului.

Observabilitatea de ordin minim pentru sistemul descris de (1) este dată de :

$$\dot{z}=Fz+Gy+hu \quad \hat{x}=Dz+Ey \tag{2}$$

unde :

$\hat{x}$ — este estimatorul lui x ;

$F=A_{22}-L$; $G=-FL+A_{21}$; $h=b_2$;

$D=(0,\ I)^T$; $E=(I,\ -L)^T$.

În situaţia concretă L este astfel determinat încît valorile proprii ale matricei F sint :

$$(-10.018,\ -17.463\pm j11.890)$$

Structura observabilităţii este prezentată în fig. 2.

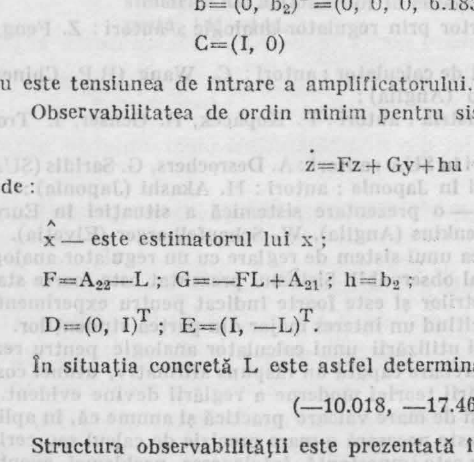


Fig. 2. Structura observabilităţii.

Pentru descrierea reacţiei inverse se utilizează teoria reglării liniare optime şi deoarece (1) este complet controlabil, pentru un timp infinit de reglare, funcţia de comandă u* care minimizează indicele de performanţă

$$J_2=\int_0^{\infty}(x^TQx+ru^2)\,dt \tag{3}$$

este dată de reacţia inversă a stării liniare

$$u^* = K^* x^*$$

Pentru uşurinţa practică s-a preferat o reacţie inversă liniară invariantă în timp ;

$$u_t = K_t x$$

unde :

$$K_t = -(1.6785,\ 25.437,\ 81.513,\ 1.8432,\ 13.298,\ 12.598)$$

Experimentul a fost încorporat în programa de instruire şi a dat bune rezultate, studenţii executînd următoarele sarcini :

1. Formularea modelului liniar care reprezintă sistemul.
2. Analiza controlabilităţii, observabilităţii, stabilităţii etc., utilizînd pachetul de programe CAD pe minicalculatorul HP-1000.
3. Descrierea regulatorului, care constă din observabilitate şi legea de reglare a reacţiei, prin utilizarea pachetului CAD.
4. Simularea sistemului de reglare prin integrare numerică.
5. Implementarea regulatorului prin intermediul circuitelor electrice şi experimentarea lui.

Activităţile 1—4 au constituit, de asemenea, diverse preexperimente şi proiecte.

Referatul 07.1—2 descrie un experiment de laborator bazat pe măsurători şi comenzi executate de un microprocesor specializat.

Suportul procesului studiat a fost o conductă de cupru, avînd la cele 2 extremităţi cîte o bobină de încălzire (Fig. 3) şi implantate de-a lungul conductei 4 traductoare de temperatură. Acest suport permite experimente multiple pentru înţelegerea proceselor cu parametri distribuiţi.

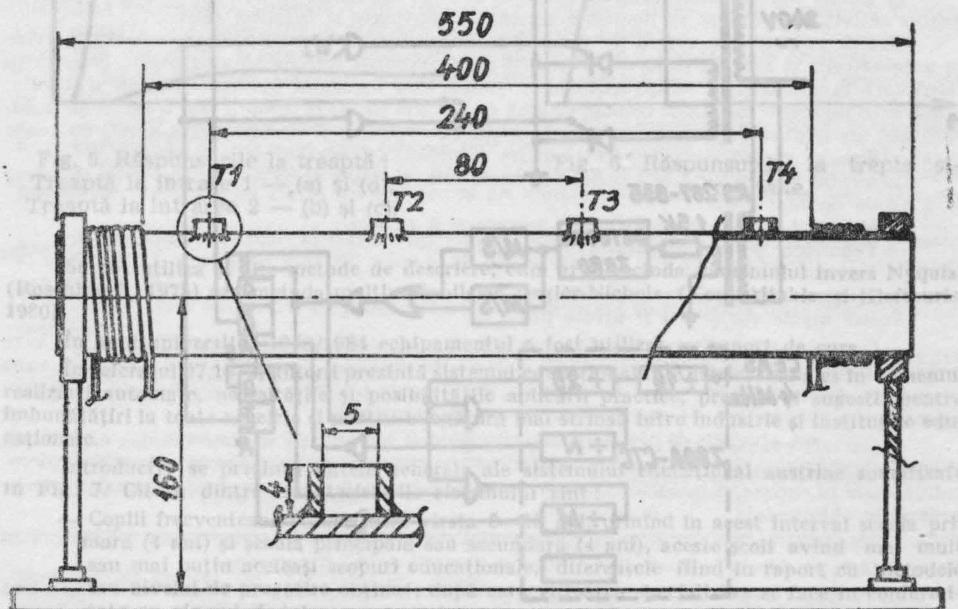


Fig. 3. Structura conductei.

Calculatorul utilizat a fost de tipul NASCOM 2, avînd la bază un microprocesor Z-80A. Configuraţia dispune de un monitor TV şi o tastatură, precum şi de o interfaţă serială cuplată la o unitate de casetă magnetică. Spaţiul de memorie este de 4 K RAM şi 4 K EPROM. Sistemul dispune de un microsoft BASIC şi de un monitor de 2 K.

Software-ul de interfeţe este ars în EPROM şi este permanent la dispoziţia utilizatorului, care realizează programarea într-un limbaj de nivel superior.

Din punctul de vedere al interfeţelor, sistemul dispune de 4 intrări în bucle de curent de la traductoarele de temperatură şi de 2 ieşiri care comandă tiristoare. Fiecare canal de intrare dispune de un convertor A/D ZN427E. Toate cele 4 conversii se execută simultan, ieşirile celor 8 biţi fiind conectate împreună într-un BUS cuplat la portul A al PIO, ceea ce reprezintă o multiplexare a semnalelor numerice şi nu a celor analogice.

Circuitele de ieşire generează întîrzieri în gama 0—10 ms, utilizînd componente countertime (CTC) din familia Z-80. Fiecare tiristor necesită 2 canale CTC şi, deoarece CTC-ul dispune de 4 canale, un CTC poate comanda ambele tiristoare. Timpul de eşantionare este generat de un al doilea CTC. Constantele de timp ale echipamentului sînt în gama 14—28 minute, iar perioada de eşantionare a fost aleasă de 15 secunde (Fig. 4).

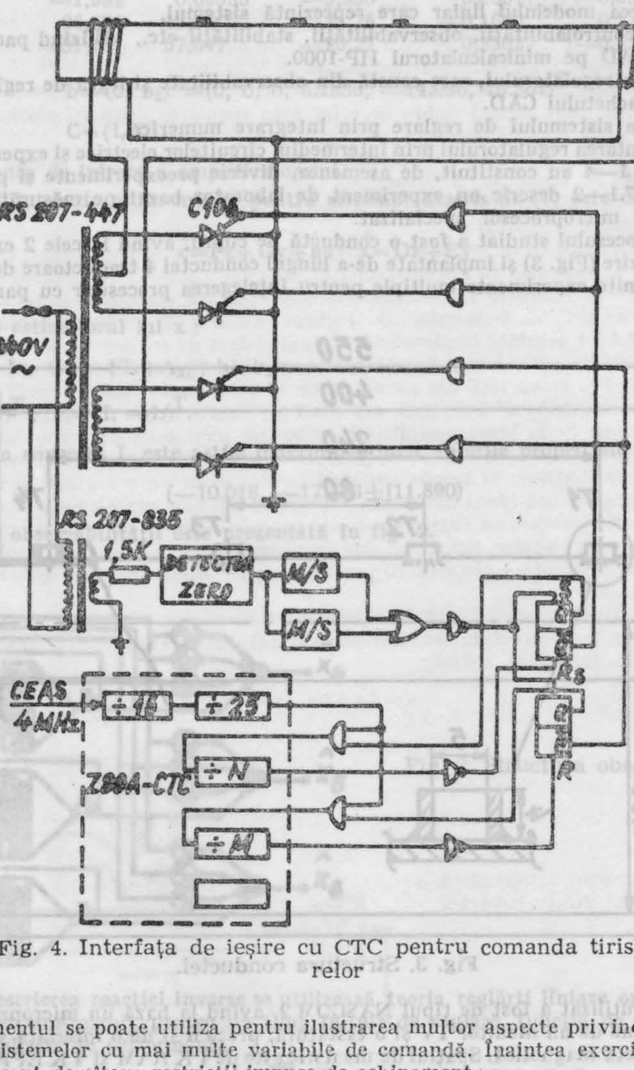


Fig. 4. Interfaţa de ieşire cu CTC pentru comanda tiristoarelor

Echipamentul se poate utiliza pentru ilustrarea multor aspecte privind descrierea şi implementarea sistemelor cu mai multe variabile de comandă. Înaintea exerciţiilor de descriere trebuie ţinut cont de cîteva restricţii impuse de echipament :

(i) Numai cîteva profiluri de temperatură pot fi accesibile ;

(ii) Variabilele de comandă au caracteristici neliniare, deoarece numai încălzirea poate fi controlată, în timp ce răcirea este determinată de construcția sistemului, ceea ce implică un grad de interacțiune în sistem, care nu poate fi eliminat ;

(iii) Modelul utilizat este aproximativ, este un model liniar, model cu parametri distribuiți ai unui sistem neliniar.

În referat se prezintă descrierea locului caracteristic (MacFarlane și Kouvaritakis, 1977). Regulatorul rezultat a constat dintr-un compensator de înaltă frecvență și un regulator comutativ de aproximare incorporînd acțiuni proporționale și integrale. Rezultatele sînt prezentate în Fig. 5 și 6.

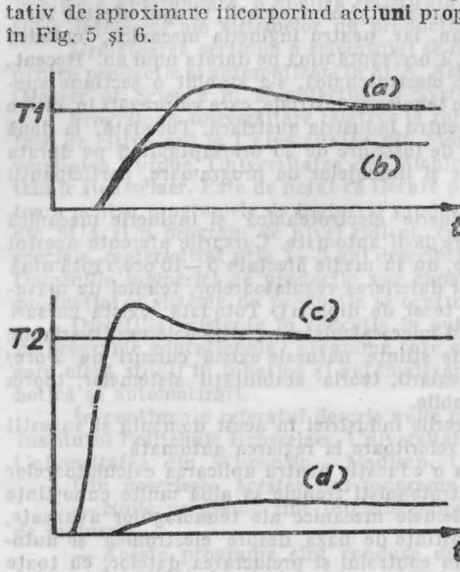


Fig. 5. Răspunsurile la treaptă :
Treaptă la intrare 1, — (a) și (d),
Treaptă la intrarea 2 — (b) și (c).

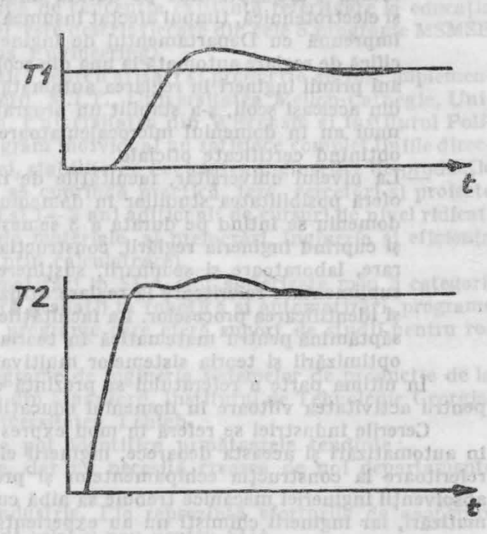


Fig. 6. Răspunsurile la trepte simultane.

Se pot utiliza și alte metode de descriere, cum ar fi metoda domeniului invers Nyquist (Rosenbrock, 1975) sau metoda multivariabilelor Ziegler-Nichols (Kouvaritakis și Kleftouris, 1980).

În anul universitar 1983/1984 echipamentul a fost utilizat ca suport de curs.

În referatul 07.1—3, autorii prezintă sistemul educațional în Austria, mai ales în domeniul realizării automate, necesitățile și posibilitățile aplicării practice, precum și sugestii pentru îmbunătățiri la toate nivelele și pentru o legătură mai strînsă între industrie și institutele educaționale.

Introductiv se prezintă datele generale ale sistemului educațional austriac sumarizate în Fig. 7. Cîteva dintre caracteristicile sistemului sînt :

— Copiii frecventează școala între vîrsta 6—15 ani, urmînd în acest interval școala primară (4 ani) și școala principală sau secundară (4 ani), aceste școli avînd mai mult sau mai puțin aceleași scopuri educaționale, diferențele fiind în raport cu metodele sau nivelul de pregătire obținut, după care viitoarea specializare se face în conformitate cu cîmpul de interes personal.

— Ca rezultat se obțin trei nivele de pregătire. La nivelul cel mai scăzut (I în Fig. 7) se găsesc școlile vocaționale cu pregătire de personal pentru industrie. La nivelul mediu (II în Fig. 7) se găsesc numeroase specializări corespunzătoare diverselor cîmpuri ale ingineriei, cursurile acestor școli, ca și ale școlii generale secundare, încheindu-se cu examen de maturitate. La nivelul superior (III în Fig. 7) se găsește universitatea tehnică, care dă posibilitatea obținerii gradului de „inginer diplomat", sau după studii adiționale a celui de „doctor în științe tehnice".

— Nivelul II poate fi atins și la colegiile tehnice (4—5 semestre), care sînt momentan în stadiul de proiect pilot. Următoarea parte a referatului prezintă situația în domeniul reglării automate pe suportul sistemului educațional. Astfel :

— În școlile vocaționale nu se predau nici un fel de noțiuni de reglare automată. Numai în ultimii ani ai școlilor generale secundare se predau noțiuni fundamentale de prelucrare electronică a datelor, dar aceste cursuri sînt opționale.

— La nivelul mediu, intensitatea educației în reglarea automată depinde de cîmpul de studiu, astfel încît pentru ingineria în electronică și comunicații, timpul afectat însumează 4 ore/săptămînă pe durata unui an, iar pentru ingineria mecanică, chimică și electrotehnică, timpul afectat însumează 2 ore/săptămînă pe durata unui an. Recent, împreună cu Departamentul de inginerie electrotehnică, s-a stabilit o secțiune specifică de reglare automată la una din școlile tehnice industriale, care va pregăti în cîțiva ani primii ingineri în reglarea automată pentru industria austriacă. Totodată, la două din aceleași școli, s-a stabilit un program de instruire de 20 ore/săptămînă pe durata unui an în domeniul microcalculatoarelor și limbajelor de programare, participanții obținînd certificate oficiale.

— La nivelul universitar, facultățile de inginerie electrotehnică și inginerie mecanică oferă posibilitatea studiilor în domeniul reglării automate. Cursurile aferente acestui domeniu se întind pe durata a 3 semestre, au în medie afectate 5—10 ore/săptămînă și cuprind ingineria reglării, construcția și descrierea regulatoarelor, tehnici de măsurare, laboratoare și seminarii, susținerea tezei de diplomă. Totodată, există cursuri suplimentare referitoare la reglare utilizînd microcalculatoare, sistemele multivariabile și identificarea proceselor. La facultățile de științe naturale există cursuri de 2 ore/săptămînă pentru matematică în teoria reglării, teoria stabilității sistemelor, teoria optimizării și teoria sistemelor multivariabile.

În ultima parte a referatului se prezintă cererile industriei în acest domeniu și sugestii pentru activitatea viitoare în domeniul educației referitoare la reglarea automată.

Cererile industriei se referă în mod expres la o educație pentru aplicarea calculatoarelor în automatizări și aceasta deoarece, inginerii electrotehniști trebuie să aibă multe cunoștințe referitoare la construcția echipamentelor și problemele mecanice ale tehnologiilor avansate, absolvenții ingineriei mecanice trebuie să aibă cunoștințe de bază despre electronică și automatizări, iar inginerii chimiști nu au experiență în controlul și prelucrarea datelor, cu toate că ei utilizează curent procedee de estimare și prelucrare statistică a datelor.

Conceptul pentru viitorul educației în acest domeniu este prezentat în Fig. 8 și se referă la 4 nivele de educație :

— Mecanici pentru dispozitive de măsurare și reglare ;

— Tehnicieni pentru măsurători și reglări ;

— Ingineri în domeniul reglării ;

— Ingineri diplomați în domeniul reglării.

Aceste nivele vor putea fi atinse prin :

— Cursuri despre dispozitive de reglare pentru specialiști în fabricația aparaturii care trebuie să cuprindă reglarea temperaturii, sisteme de ventilare și condiționare a aerului, vane de reglare, dispozitive de reglare pneumatice, regulatoare programabile, regulatoare cu microcalculatoare, dispozitive de manipulare și roboți industriali.

— Cursuri avansate de ingineria reglării, pentru absolvenții școlilor tehnice certificate, despre comandă numerică, sisteme de reglare multivariabile, procese dinamice și modele, microcalculatoare în reglare, dispozitive de manipulare și roboți industriali, montaj automat, reglarea proceselor chimice.

— Lucrul la un proiect concret în domeniul reglării pentru o întreprindere sau un centru de cercetări, pe durata a 6—8 săptămîni, în grupuri de 4—5 persoane.

În referatul 07.1—4, „Tendințe educaționale în automatizări în SUA", autorii prezintă cîteva eforturi recente în elaborarea programelor pentru ingineria sistemelor de producție, în cîteva universități mari din SUA și susțin că aceasta este una din căile de cooperare între universitățile publice și industria privată.

În introducere autorii pornesc de la faptul că din 1960 pînă în 1970, față de producția unui anumit sector, productivitatea muncii a fost în scădere și subliniază că una din cauze este existența unei conduceri tehnice nepregătită pentru producție. În sprijinul acestui fapt ei aduc argumentul că, în prezent, din 1216 programe acreditate de inginerie în SUA, numai 3 sînt programe de inginerie de producție.

O soluţie este modificarea pregătirii educaţionale, mai ales în domeniile în care tehnologia automatizării poate aduce un efect important privind creşterea productivităţii muncii (Tabelul 1).

Necesităţile pentru noile programe de ingineria producţiei trebuie să pornească de la schimbările rapide în tehnologia calculatoarelor, microprocesoarelor, traductoarelor, roboţilor şi inteligenţei artificiale. Aceste schimbări rapide vor crea un nou profil de inginer de producţie, astfel încît dezvoltarea şi individualizarea funcţiilor în cadrul acestor evenimente trebuie să implice o aprofundare, o analiză sofisticată şi o prelucrare atentă, pentru înţelegerea implicaţiilor intrinseci ale sistemelor de producţie.

Următoarea secţiune a referatului se ocupă de concepţia existentă referitoare la educaţia pentru producţie şi automatizări în SUA. Astfel, este descris fiecare din cele 5 programe MSMSE (Master of Science in Manufacturing Systems Engineering).

Programele universitare relative la robotică, automatizări şi producţie au fost implementate de cîţiva ani la Universitatea din Stanford, M.I.T., Universitatea Mellon-Carnegie, Universitatea Purdue, Universitatea din Michigan, Universitatea din Wisconsin şi Institutul Politehnic Rensselaer. Este de notat că fiecare program individual nu satisface complet liniile directoare pentru programele de ingineria producţiei, stabilite de Societatea inginerilor de producţie din SUA (Un program de nivel ridicat, care să cuprindă practică, teze, cercetări şi proiecte speciale, materializat prin 1—3 ani descriptivi şi 1—3 ani adiţionali de cursuri de nivel ridicat. Domeniile ingineriei de producţie vor cuprinde materiale şi prelucrări, ingineria şi eficienţa producţiei şi sisteme de producţie şi control prin calculatoare).

Direcţiile existente în educaţia în automatizări în SUA sînt sintetizate prin 3 categorii de programe educaţionale : programe care oferă gradaţii în robotică şi automatizări, programe care oferă studii în robotică şi automatizări, programe care oferă suport de studii pentru robotică şi automatizări.

În continuare referatul descrie noile programe de ingineria sistemelor de producţie de la Institutul Politehnic Rensselaer, Universitatea din Standford, Institutul de Tehnologie Georgia, Universitatea din Wisconsin — Madison, Universitatea Lehigh.

Din descrierea acestor noi programe se pot identifica următoarele tendinţe :

— Noile programe sînt interdisciplinare, dar nu necesită crearea de noi departamente de inginerie.

— Aceste programe sînt conduse de industrie. Ele reprezintă eforturile de cooperare între universităţi şi industrie, care este un fapt relativ nou pentru SUA.

— Noile cercetări pun accentul pe partea practică a pregătirii, în ideea trecerii de la un inginer de producţie cu statut de artizan, la un inginer de producţie orientat larg către producţie.

— Un efort continuu şi clar se întreprinde pentru strîngerea legăturilor între industrie şi universităţi, prin programe de schimb de vizite reciproc, un fapt care este, de asemenea, nou pentru SUA.

— Ingineria sistemelor de producţie a fost identificată ca existentă între inginerie şi conducere.

— Aceste programe sînt labile şi tind să ducă la un răspuns rapid privind realinierea conceptelor de proiectare şi producţie.

Tabelul 1

Programe de creştere a productivităţii muncii şi eficienţa lor

Tipul de program	Eficienţa raportată a programelor		
	Ridicată	Medie	Scăzută
Investiţii de capital sau automatizări (fără roboţi)	48 %	45 %	8 %
Sisteme inovatoare (unităţi integrate, tehnici de producere de materiale)	37 %	49 %	14 %
Programe de stimulare a salariilor	35 %	41 %	24 %
Introducerea şi extinderea utilizării roboticii	32 %	34 %	34 %
Introducerea sau dezvoltarea metodelor de control	30 %	53 %	18 %

În referatul 07.1—5, „Tendinţe în educaţia în automatizări în Japonia", autorul prezintă rezultatul unui sondaj de opinie, împreună cu un raport referitor la activitatea cîtorva laboratoare. Concluzia este că, unitar, situaţia generală pare cam conservatoare, dar activitatea este foarte bine organizată.

În partea introductivă se prezintă sistemul învăţămîntului superior, pe baza cîtorva statistici comparative cu SUA, prezentate în Tabelul 2. Următoarele elemente sînt caracteristice:

— Proporţia între numărul universităţilor şi colegiilor pe întreaga populaţie este de 1 universitate la 116 500 oameni în Japonia, în timp ce în SUA este de 1 universitate la 67 909 oameni.

— Numărul studenţilor admişi la facultate în Japonia este aproximativ egal cu numărul studenţilor care o absolvă (aprox. 80 %).

— Numărul studenţilor în inginerie este aproximativ 1/5 din total. Dintre aceştia 1/20 sînt înscrişi pentru titlul Master şi 1/200 pentru titlul Doctor.

— Valoarea cheltuielilor în învăţămînt a sporit mult în Japonia începînd cu 1975 (Tabelul 3).

O deosebire principală apare la sistemul de admitere în învăţămîntul superior, unde, spre deosebire de universităţile americane şi europene, se organizează practic două concursuri de admitere pentru acelaşi candidat. Totodată, datorită acestui sistem de admitere, este mai uşor să intri în facultate decît să o termini, deoarece mulţi din cei care au trecut grelele examene de admitere consideră că munca lor a luat sfîrşit.

În continuare referatul prezintă sistemul educaţional în automatizări. Se observă că deşi ingineria automatizărilor este relativ recentă în învăţămîntul universitar, ea a fost predată în multe universităţi, sub diverse aspecte, începînd cu sfîrşitul ultimului război mondial.

În cazul Universităţii din Kyoto, studiile speciale de sisteme de reglare sînt întotdeauna popularizate şi adeseori studenţii de anul 4 concurează în asociere cu laboratoarele de inginerie pentru automatizări, pentru susţinerea tezei de diplomă.

Numărul de ore pentru experimentări este foarte mare (Tabelul 4) şi se pare că multe departamente folosesc calculatoare analogice şi servomecanisme. Totuşi, numărul microcalculatoarelor utilizate este în creştere rapidă.

Subiectele predate şi numărul orelor afectate pentru automatizări sînt prezentate în tabelul 5.

În ceea ce priveşte utilizarea microcalculatoarelor din cele 116 departamente interogate, 101 utilizează microcomputere de diverse tipuri şi capacităţi (Tabelul 6). Majoritatea acestor microcomputere se utilizează pentru măsurători, prelucrări de date, analiză numerică, simulare şi ca terminale pentru calculatoarele puternice. Şase departamente utilizează aceste microcalculatoare în comanda roboţilor.

Din cele prezentate rezultă cîteva trăsături esenţiale:

— Predarea se face în domeniile tradiţionale (mecanică şi electrotehnică) şi constă în majoritate în teorie fundamentală.

— Avînd în vedere indecizia între mai multă teorie sau mai multă practică şi opţiunile unora pe partea neconvenţională se ajunge la concluzia unei necesităţi a unirii teoriei cu practica.

— Lipsa de fonduri bugetare şi deci prea puţine ore de predare, precum şi lipsa de cadre competente duc la o ineficienţă a pregătirii, la un gol între teorie şi problemele actuale.

În finalul referatului, utorul prezintă tendinţele în educaţia în automatizări pe baza unor cazuri specifice. Concluziile trase s-ar putea rezuma la:

— Extinderea teoriei controlului şi reglării prin hardware. Aceasta poate avea implicaţii directe asupra sistemelor CAD şi în special asupra elaborării de software în timp real.

— Necesitatea predării teoriei moderne a reglării chiar de la început, dacă se are în vedere că metodologia de descriere lipseşte din teoria clasică a reglării.

— Necesitatea extinderii utilizării microcalculatoarelor pentru studiul reglării automate pentru calcul). De asemenea, teoria modernă a reglării reclamă neapărat utilizarea microcalculatoarelor (rezolvarea matricială a ecuaţiilor Ricatti, problemele legate de sistemele discrete, răspunsul treaptă, impuls, diagrama Bode, locul rădăcinilor necesită un software adecvat modelele stochastice sintetizate uşor, producerea numerelor aleatoare după o distribuţie statistică, filtrele Kalman).

— Utilizarea cît mai intensă a microcalculatoarelor în robotică.

În referatul 07.1—6, „Orizonturi în educația inginerească — o prezentare sistemică a situației în Europa de vest“, autorii utilizează metodele ingineriei sistemelor pentru a investiga sistemul educațional ingineresc și a-i deduce orizonturile.

Introducerea pornește de la aprecierea că, pe baza unor cercetări ale unui grup de lucru SEFI (Société Européene de la Formation des Ingienieurs), sistemul de educație tehnică în Europa de vest este într-o stare de fapt periculoasă. Autorii utilizează metodele teoriei sistemelor pentru ca punctele subiective de vedere să devină mai obiective.

Sistemul este prezentat în Fig. 9 și se caracterizează prin:

— Sistemul a fost conceput pentru a fi ușor urmat de ingineri și nu de mecanici sau tehnicieni. Sistemul a fost conceput în afara științei și suferă din această cauză.

— Natura ieșirilor sistemului este în principal determinată de sistem însuși, el nerăspunzînd presiunilor din afară.

— Presiunile interne sînt concentrate spre introducerea rezultatelor cercetărilor în știință și inginerie.

— Sistemul conține propriile sale metode de studiu.

— Sistemul nu are o preocupare particulară pentru lărgirea orizontului, se autoperpetuează, produce personal pentru educație și în consecință, are un orizont închis.

— Scala timpului pentru învățare este relativ lungă, ciclul de dezvoltare al sistemului este de durată.

— Adeseori se spune că răspunde la feedback, dat timpul de răspuns este așa de mare, încît el tinde să fie un sistem foarte stabil.

— Este foarte puțin receptiv la schimbare, nu este un sistem auto-motivabil.

— Dacă sîntem de acord cu aceste puncte vom vedea clar clasificarea sistemului educațional, ca nu sistem, dar nu ca un sistem de succes.

Pentru îmbunătățirea sistemului, autorii propun două soluții, modelul elementelor organizaționale și procesul de rezolvare al problemei în șase pași.

Modelul elementelor organizaționale constă în stabilirea următoarelor:

— Intrări (resurse materiale)	resurse umane și fizice, decizii guvernamentale, legi, necesități, obiective, fonduri
— Procese (cum se manifestă acestea)	cursuri, programe, cercetări, metode de studiu și tehnologii
— Produse (rezultate implementate)	cursuri și proiecte complete, servicii furnizate către utilizatori
— Ieșiri (produse agregate ale unei organizații care sînt furnizate către societate)	licențe, certificate personale, rezultatele cercetărilor, publicații.

În partea finală a referatului, autorii încearcă să prezinte tendințele de viitor prin prisma a două categorii de elemente.

Prima categorie de elemente constituie factorii de influență care sînt grupați după:

I. Necesitățile industriei;
Efectele schimbărilor tehnologice în societate;
Acceptarea de către societate;
Reacții împotrivă.

II. Evoluția însăși a ingineriei;
Evoluția comunicațiilor și informaticii;
Evoluția noilor tehnologii.

III. Efectele sociale:
Schimbările de cerințe asupra profesiunii de inginer;
Schimbările punctelor de vedere tradiționale asupra ingineriei.

A doua categorie de elemente o constituie direcțiile pentru viitor:

I. Cunoștințele profesionale;
Tehnologia de lucru;
Natura noilor cunoștințe și aplicarea lor.

II. Natura societății;
Etica de lucru și modificarea ei;
Echilibrarea între timpul de muncă și odihnă;
Efectele cibernetizării și informatizării.

III. Natura ingineriei ;
Cum va fi ea definită în viitor ;
Care sînt aspectele modificărilor inclusiv în sistemul de învăţămînt.
IV. Categoriile de ingineri în viitor ;
Ce cunoştinţe acumulează ;
Ce munci li se vor da ;
Ce responsabilităţi vor avea ;
Punctul de vedere asupra profesionalismului la sfîrşitul secolului.
V. Natura cunoştinţelor cerute pentru aceste categorii de ingineri ;
Subiectele tradiţionale opuse noilor subiecte ;
Modul în care ele vor fi furnizate în şcoală ;
Efectele conservării energiei, concentrării industriei ;
Atitudinea în şcoală.
VI. Natura industriei pentru societate.

Concluzia referatului este că sistemul de educaţie inginerească trebuie schimbat, dar utilizînd metoda bottom-up, care necesită un timp mai scurt şi modificări mai mici, faţă de metoda top-down.

II) CS.5 SISTEMUL EDUCAŢIONAL DIN R.P. UNGARIA ŞI TRANSFERUL SĂU PESTE HOTARE *

Sistemul educaţional din R.P. Ungaria nu este prea bine cunoscut în lume. Prezenta expunere îşi propune să facă o prezentare generală a condiţiilor în care se desfăşoară educaţia în R.P. Ungaria şi a sistemului în care sînt pregătiţi inginerii. Se prezintă organizarea Universităţii Tehnice din Budapesta (cea mai mare instituţie de învăţămînt superior din R.P.U) precum şi a Facultăţii de Electrotehnică. În continuare se fac consideraţii asupra exportului de sisteme complete de educaţie.

Ungaria are cca 11 milioane locuitori. Puterea creatoare a poporului este demonstrată de o gamă largă de realizări deosebite. Şase savanţi de origine maghiară sînt laureaţi ai Premiului Nobel : Richard Zsigmond (chimie) în 1925, Albert Szentgyörgyi (medicină) în 1937, György Hevesi (chimie) în 1943, György Békési (medicină) în 1961, Jenő Wigner (fizică) în 1963 şi Dénes Gábor (fizică) în 1971.

Dintre oamenii de ştiinţă maghiari de faimă mondială fac parte János Bòlyai, Ignácz Semmelweis, Loránd Ötvös, János Neumann, Tòdor Kálmán şi Leo Szilárd. În lumea muzicii sînt faimoase lucrările lui Liszt, Lehár, Bartôk, Kodaly şi mulţi alţi compozitori.

Sistemul educaţional din Ungaria cuprinde :

Învăţămîntul obligatoriu, care începe la vîrsta de şase ani cu *Şcoli primare*, în care fiecare copil primeşte pînă la vîrsta de 14 ani, acelaşi tip de educaţie.

Urmează *şcolile pentru pregătire profesională*, în care se pregătesc cadre pentru cca 255 de profesiuni, cu o durată de 3 sau uneori de 2 ani.

Şcolile gimnaziale sînt tipul de şcoală cu cea mai veche tradiţie.

După absolvirea cu succes a celor patru ani de şcoală gimnazială, elevii se pot prezenta la examenul de maturitate.

Şcolile secundare de specialitate experimentale au început să funcţioneze începînd cu anul 1959 şi ca rezultat s-a născut o şcoală de tip nou.

Orice cetăţean al R.P.U, pînă la vîrsta de 35 de ani poate solicita admiterea în învăţămîntul superior, cu condiţia absolvirii examenului de maturitate.

Universitatea tehnică din Budapesta îşi începe istoria în jurul anului 1782 ; denumirea actuală datează din anul 1949. Ea este cea mai mare universitate din ţară şi este singura care are o facultate de electrotehnică.

Întregul corp profesoral al universităţii numără cca 2000 de cadre didactice. Biblioteca Universităţii Tehnice dispune de aproape o jumătate de milion de volume de specialitate.

Universitatea dispune de un calculator electronic R32 de fabricaţie poloneză. În plus, fiecare facultate are un calculator propriu CDPA polonez şi minicalculatoare SM-4 de fabricaţie

---

* Z. BENYÓ, Catedra de Automatică, Universitatea Tehnică din Budapesta Facultatea de Electrotehnică — R.P. Ungaria. *(Material adaptat).*

sovietică. Facultatea de Electrotehnică dispune și de un calculator electronic IBM 370/115 și mai multe minicalculatoare de tip PDP-11.

Universitatea Tehnică din Budapesta are cca 8 000 de studenți.

Facultatea de Electrotehnică de la Universitatea Tehnică din Budapesta a fost înființată în anul 1959. Ca urmare a dezvoltării sale deosebit de dinamice, facultatea are în prezent cca 2 200 studenți și 350 cadre didactice.

Exportul R.P. Ungare în domeniul educației se bazează pe accelerarea diviziunii internaționale a muncii de după cel de al 2-lea război mondial, cînd schimbul de servicii și de experiență intelectuale, alături de schimburile de bunuri materiale, a devenit din ce în ce mai important.

Specialiștii noștri de înaltă calificare activează în țări în curs de dezvoltare din Africa, Asia și într-o oarecare măsură și în America de Sud. Mai recent, au crescut cerințele pentru experți în agricultură.

O altă formă de cooperare cu țările în curs de dezvoltare este exportul de clase complete pentru învățămînt. Este vorba de clase, laboratoare, ateliere de învățămînt complet echipate, livrarea echipamentelor necesare, punerea la dispoziție a materialului didactic și a suporturilor de curs, trimiterea de personal didactic și ajutător precum și pregătirea personalului didactic al țării importatoare, fie în Ungaria fie în țara respectivă.

Pentru a realiza o instruire adusă la zi, s-a înființat și un Institut pentru educație și metodologia educației. Ministerul Educației îndrumă instituțiile care elaborează programele analitice, cerințele, metodele și mijloacele necesare educației publice. Asemenea instituții sînt Institutul Pedagogic Național, Centrul Național pentru tehnici de instruire, întreprinderile ce produc material didactic, Institutul de artă populară etc.

Instruirea în domeniul agriculturii și industriei alimentare are o situație specială, ea depinzînd direct de Ministerul Agriculturii și Industriei Alimentare. Un Centru de instruire și de cercetare precum și un Institut metodologic și de studii post-universitare contribuie nemijlocit la această instruire și la stabilirea cerințelor.

Întreprinderile mai importante din R.P.U. care realizează export de facilități pentru instruire și expertiză profesională sînt : TESCO, KOMPLEX, TECHNOIMPEX, METRIMPEX, INDUSTRIALEXPORT, KÖZTI, LABOR MIM, AGTI. Aceste întreprinderi sînt toate întreprinderi de stat ; în consecință, produsele și serviciile contractate sînt garantate în întregime de către stat.

Fără a avea pretenția că lista este completă, iată cîteva din realizările noastre :

Țara	Denumire și tip proiect	Caracteristici
1	2	3
Libia	Institut pentru calificarea muncitorilor pentru 500 persoane	Pentru calificarea de specialiști în industria lemnului, livrare echipamente instalare și punere în funcțiune
Peru	Diferite laboratoare de instruire	Pentru fizică, metalografie analitică, mecanică electrochimică, biologie
Algeria	Universitatea de științe și tehnologie din Oran	Livrarea de echipamente pentru laboratoare și ateliere pentru Institutul de Arhitectură și de Inginerie Mecanică. Elaborarea de echipamente auxiliare pentru instruire, suporturi de curs în lb. franceză. Proiectare, conducere tehnică, asistență tehnică și service
Iran	Institut pentru calificarea muncitorilor	Livrarea și instalarea de echipamente pentru laboratoare și ateliere necesare predării de cunoștințe în domeniile industriei alimentare, chimice, metalurgie, electrotehnică
Nigeria	Politehnica Federală din Idah	Planificarea pedagogică ; pregătirea suporturilor de curs, îndrumarele de practică ; livrarea de echipamente și instalarea lor pentru laboratoare și ateliere pentru facultățile de arhitectură, electrotehnică, mecanică și metalurgie

1	2	3
Nigeria	Politehnica Federală din Yola	Livrarea de echipamente pentru laboratoare și ateliere ; instruirea cadrelor didactice, documentare pedagogică, instruire în R.P.U., asistență tehnică în Nigeria
Nigeria	Institutul de calificare a muncitorilor, Colegiul Tehnic	Livrarea de laboratoare și ateliere necesare pentru instruirea în diferite profesii
Nigeria	Politehnica de Stat din Ogun	Livrarea de laboratoare, instruire în R.P.U. și asistență tehnică
Irak	Institute de calificare a muncitorilor pentru mecanizarea agriculturii	Proiect „la cheie" pentru clădire, livrare și instalare echipament didactic Pregătirea instructorilor în R.P.U.
Cuba	Diferite laboratoare de instruire	Livrare echipamente pentru laboratoare și ateliere universitare.
Bangladesh	Laborator de instruire	Livrare echipamente de laborator
Mexic	Laboratoare de instruire și de testări pentru industria alimentară	Livrarea și instalarea de laboratoare complete
Nigeria	Fabrică de echipamente de cercetare	Proiectare și realizare
Algeria	Școli secundare de specializare în hidrologie la : Banchegout Kaar — Chellala Biskra M' Sila	Proiectarea construcției, livrarea și instalarea de echipamente
Brazilia	Livrarea de echipamente și instrumente pentru universități	
Vietnam	Livrarea de echipamente pentru o școală de specializare a 1.000 de studenți, Hanoi	

BIBLIOGRAFIE

Benyó, Z. : Educational system of Hungary professional education, educational export in Hungary, *Komplex-Symposium,* Cairo 28—29 nov., 1982.

Benyó, Z. : Programmable teaching equipments. Experiences of professional training in Hungary. *Asean Didacta 83',* Singapore 7—11 June, 1983.

Cheng, D.K. : Electrical Engineering Education in Hungary. *IEEE transaction on education,* Vol. E—17, No. 2, may 1974, pag. 92—99.

Szabadváry, F. : Budapest Technical University, 1782—1982 *Periodica polytechnica,* 1982.

Vágó, I. : Budapest Technical University, Faculty of Electrical Engineering, 1982.

SISTEME COMPLEXE ŞI MARI ; TEORIE, METODOLOGIE ŞI APLICAŢII

Ing. *Laurenţiu Orăşanu*
Dr. ing. *Florin Gheorghe Filip*
Ing. *Melia Muratcea*
I.T.C.I.

Lucrările grupate în cele 5 secţiuni ale *colocviului 11.1.* al Congresului 9 IFAC ilustrează — prin numărul lor (29) şi diversitatea problematicii abordate — interesul constant al specialiştilor pentru găsirea unor noi abordări teoretice şi pentru lărgirea gamei de aplicaţii în conducerea sistemelor complexe.

1. ANALIZA ŞI DESCOMPUNEREA SISTEMELOR COMPLEXE

În sensul teoriei introduse pentru prima dată în [1] şi [2], *lucrarea 11.1/A1* prezintă două metode de măsurare exactă a complexităţii, una bazată pe informaţia conţinută, alta pe aspecte de calcul, cu aplicaţii în prelucrarea datelor, teoria probabilităţii şi teoria algoritmilor. Pentru a introduce noţiunea de complexitate (Kolmogorov) a informaţiei se consideră datele şi programele ca secvenţe de 0 şi 1 (pe scurt şiruri). Fie X un şir şi $|X|$ lungimea sa. Se zice că un şir Y defineşte pe X dacă Y este un program (BASIC) care la execuţie tipăreşte pe X. Complexitatea Kolmogorov K (X) a şirului X este definită ca lungimea minimă a oricărui şir care defineşte pe X. Cel mai scurt program care defineşte pe X este codul Kolmogorov al lui X. Pentru alt program BASIC sau alt limbaj se obţine o valoare K′ (X) diferită

$$|K(X) - K'(X)| < \text{const.} \tag{1}$$

n care constanta depinde numai de cele 2 limbaje, nu de X.

Complexitatea Kolmogorov măsoară conţinutul în informaţie al şirului. Dacă un şir conţine aproximativ $\alpha |X|$ de 1 $(0 < \alpha < 1)$, atunci şirul are complexitatea informaţiei cel mult $H(\alpha)|X|$, unde:

$$H(\alpha) = -\alpha \log_2 \alpha - (1-\alpha) \log_2 (1-\alpha) \tag{2}$$

În ceea ce priveşte complexitatea de calcul, se abordează aspectul timp de calcul. Pentru un program operînd cu o intrare dată X se compară de obicei timpul de execuţie cu lungimea $|X|$ a intrării. Astfel complexitatea de timp a unui algoritm este definită ca o funcţie f, unde f (n) este timpul de calcul maxim al unui algoritm pentru toate intrările de mărime n. Definind o problemă ca o mulţime de şiruri, lucrarea pune în evidenţă diferite clase de complexitate a problemelor (probleme fără algoritm — axiome matematice, probleme rezolvabile în timp exponenţial, în timp polinomial, în timp linear).

În concluzie, lucrarea 11.1/A1 propune o teorie matematică a complexităţii care să poată fi utilizată în analiza şi modelarea sistemelor complexe. Pentru fiecare model ar trebui studiată atent complexitatea datelor care constituie modelul şi complexitatea de calcul a algoritmilor cu care va fi rezolvată problema.

Analiza stabilităţii sistemelor complexe (cazul discret) constituie obiectul *lucrării 11.1/A2*. Există două metode de analiză a stabilităţii sistemelor complexe bazate pe funcţiile Liapunov :
1° metoda funcţiei scalare Liapunov ;
2° metoda funcţiei vectoriale Liapunov.

Prima metodă prezintă avantajul că domeniile de stabilitate pot fi determinate foarte rapid, ceea ce o recomandă ca o metodă eficientă în rezolvarea problemelor tehnice. Metoda a fost frecvent utilizată [3], [4], [5] în diverse cazuri: sisteme invariante sau variante în timp, sisteme cu întîrzieri, continue sau discrete. A doua metodă, ca şi principiul de comparaţie (prezentat în lucrare), conduc la condiţii mai „slabe" (adică un domeniu de stabilitate mai larg) decît în cazul utilizării primei metode. În lucrare se arată că proprietăţile de stabilitate ale unui sistem complex de ordinul n pot fi obţinute prin analiza stabilităţii unui sistem de comparaţie de ordinul r ($1 \leqslant r \leqslant n$). Acest principiu de comparaţie enunţat în [6] este extins în lucrarea menţionată pentru cazul sistemelor discrete în timp.

În lucrarea *1.1/A3* este extins principiul de incluziune pentru conducerea descentralizată cu suprapunerea stărilor pentru sistemele interconectate (subsistemele au anumite intrări şi ieşiri în comun). Cînd un sistem este compus din subsisteme interconectate, spaţiul stărilor este extins într-un spaţiu mai mare printr-o transformare lineară (singulară), astfel că sistemele apar ca disjuncte. Dacă extensia este realizată astfel încît sistemul extins include sistemul iniţial, se pot analiza stabilitatea şi optimalitatea sistemului extins utilizînd tehnicile standard de descompunere disjunctă. Legea de comandă proiectată în spaţiul extins poate fi „comprimată" pentru implementarea în sistemul iniţial.

Se consideră sistemele S şi $\tilde{S}$:

$$S: \dot{x}=Ax+Bu \qquad y=Cx \tag{3}$$

$$\tilde{S}: \dot{\tilde{x}}=\tilde{A}\tilde{x}+\tilde{B}\tilde{u} \qquad \tilde{y}=\tilde{C}\tilde{x} \tag{4}$$

şi perechile de transformări lineare (singulare) între stările, intrările şi ieşirile lor:

$$\begin{array}{lll} \tilde{x}=Vx & x=U\tilde{x} & UV=I_n \\ \tilde{u}=Ru & u=Q\tilde{u} & QR=I_m \\ \tilde{y}=Ty & y=S\tilde{y} & ST=I_p \end{array} \tag{5}$$

Principiul de incluziune extins: Se spune că sistemul $\tilde{S}$ include sistemul S (sau S este inclus în $\tilde{S}$) dacă există perechile de matrici (U, V) şi (R, S) astfel încît, pentru orice stare iniţială x_0 şi orice intrare fixată u (t) a lui S, alegerea stării iniţiale $\tilde{x}_0$ şi a intrării $\tilde{u}$ (t) a lui $\tilde{S}$:

$$\begin{array}{ll} \tilde{x}_0=Vx_0 & \\ \tilde{u}(t)=Ru(t) & \text{pentru } t\geqslant 0 \end{array} \tag{6}$$

implică:

$$x(t;\, x_0,\, u)=U\tilde{x}(t;\, \tilde{x}_0,\, \tilde{u})$$

$$y[x(t)]=S\tilde{y}[\tilde{x}(t)] \qquad \text{pentru } t\geqslant 0 \tag{7}$$

Condiţia necesară şi suficientă pentru incluziunea extinsă este:

$$\begin{array}{lll} A^i=U\tilde{A}^iV & A^iB=U\tilde{A}^i\tilde{B}R & \\ CA^i=S\tilde{C}\tilde{A}^iV & CA^iB=S\tilde{C}\tilde{A}^i\tilde{B}R & i=0,1,2\ldots \end{array} \tag{8}$$

Cînd considerăm descompunerea cu suprapunere a lui S, dacă perechile de matrici de transformare (U, V), (Q, R), (S, T) sînt specificate, atunci matricile A, B, C ale sistemului extins $\tilde{S}$ sînt date de:

$$\tilde{A}=VAU+M \qquad \tilde{B}=VBQ+N \qquad \tilde{C}=TCU+L \tag{9}$$

unde M, N, L sînt matrici complementare de dimensiuni corespunzătoare. Alegerea matricilor M, N, L astfel ca $\tilde{S}$ să fie o extensie a lui S se poate realiza astfel:

$$\begin{array}{lll} UM^iV=0 & UM^{i-1}NR=0 & \\ SLM^{i-1}V=0 & SLM^{i-1}NR=0 & i=\overline{1,\tilde{n}} \end{array} \tag{10}$$

Lucrarea formulează legea de comprimare sub forma : Legea de comandă $\tilde{u}=\tilde{K}\tilde{x}+\tilde{V}$ pentru extinderea $\tilde{S}$ este comprimabilă în legea de comandă $u=Kx+V$ pentru sistemul (iniţial) S dacă :

$$\tilde{x}_0=Vx_0$$

$$\tilde{u}(t)=Ru(t) \qquad (t\geqslant 0) \tag{11}$$

implică :

$$\tilde{K}\tilde{x}(t\,;\,\tilde{x}_0,\,\tilde{u})=RKx(t\,;\,x_0,\,u) \qquad (t\geqslant 0) \tag{12}$$

pentru orice stare iniţială x_0 şi orice intrare fixată u (t) a lui S.

Definind matricea F prin $\tilde{K}=RKU+F$, condiţiile necesare şi suficiente de comprimabilitate sînt date de :

$$RKA^i=\tilde{K}\tilde{A}^iV,\; RKA^iB=\tilde{K}\tilde{A}^i\tilde{B}R \qquad i=0,1,2\ldots \tag{13}$$

sau echivalent :

$$FM^{i-1}V=0 \qquad FM^{i-1}NR=0 \qquad i=\overline{1,\tilde{n}} \tag{14}$$

În consecinţă, proiectînd legea de comandă pentru sistemul extins şi aceasta fiind comprimabilă se poate obţine legea de comandă pentru sistemul iniţial pe baza :

$$K=Q\tilde{K}V \tag{15}$$

Pentru exemplificare se consideră sistemul $S:\dot{x}=Ax+Bu$ de forma :

$$S:\begin{vmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\end{vmatrix}=\begin{vmatrix}A_{11} & A_{12} & A_{13}\\ A_{21} & A_{22} & A_{23}\\ A_{31} & A_{32} & A_{33}\end{vmatrix}\begin{vmatrix}x_1\\ x_2\\ x_3\end{vmatrix}+\begin{vmatrix}B_{11} & 0 & 0\\ 0 & B_{22} & 0\\ 0 & 0 & B_{33}\end{vmatrix}\begin{vmatrix}u_1\\ u_2\\ u_3\end{vmatrix} \tag{16}$$

Sistemul S poate fi privit ca un sistem interconectat, compus din 2 subsisteme cu starea x_2 suprapusă :

$$S_1:\begin{vmatrix}\dot{x}_1\\ \dot{x}_2\end{vmatrix}=\begin{vmatrix}A_{11} & A_{12}\\ A_{21} & A_{22}\end{vmatrix}\begin{vmatrix}x_1\\ x_2\end{vmatrix}+\begin{vmatrix}B_{11} & 0\\ 0 & B_{22}\end{vmatrix}\begin{vmatrix}u_1\\ u_2\end{vmatrix} \tag{17}$$

$$S_2:\begin{vmatrix}\dot{x}_2\\ \dot{x}_3\end{vmatrix}=\begin{vmatrix}A_{22} & A_{23}\\ A_{32} & A_{33}\end{vmatrix}\begin{vmatrix}x_2\\ x_3\end{vmatrix}+\begin{vmatrix}B_{22} & 0\\ 0 & B_{33}\end{vmatrix}\begin{vmatrix}u_2\\ u_3\end{vmatrix}$$

În extinderea $\tilde{S}$, S_1 şi S_2 apar ca disjuncte :

$$\tilde{S}:\begin{vmatrix}\dot{\tilde{x}}_1\\ \dot{\tilde{x}}_2\end{vmatrix}=\begin{vmatrix}A_{11} & A_{12} & 0 & A_{13}\\ A_{21} & A_{22} & 0 & A_{23}\\ A_{21} & 0 & A_{22} & A_{23}\\ A_{31} & 0 & A_{32} & A_{33}\end{vmatrix}\begin{vmatrix}\tilde{x}_1\\ \tilde{x}_2\end{vmatrix}+\begin{vmatrix}B_{11} & 0 & 0 & 0\\ 0 & B_{22} & 0 & 0\\ 0 & 0 & B_{22} & 0\\ 0 & 0 & 0 & B_{33}\end{vmatrix}\begin{vmatrix}u_1\\ u_2\end{vmatrix} \tag{18}$$

comenzile descentralizate fiind date de :

$$\tilde{u}_1=\begin{vmatrix}\tilde{K}_{11} & \tilde{K}_{12}\\ \tilde{K}_{21} & \tilde{K}_{22}\end{vmatrix}\tilde{x}_1 \qquad \tilde{u}_2=\begin{vmatrix}\tilde{K}_{33} & \tilde{K}_{34}\\ \tilde{K}_{43} & \tilde{K}_{44}\end{vmatrix}\tilde{x}_2 \tag{19}$$

Matricea de transfer pentru sistemul extins $\tilde{S}$:

$$\tilde{K}_1 = \begin{vmatrix} \tilde{K}_{11} & \tilde{K}_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & \tilde{K}_{33} & \tilde{K}_{34} \\ 0 & 0 & \tilde{K}_{43} & \tilde{K}_{44} \end{vmatrix} \tag{20}$$

nu este comprimabilă la :

$$K = \begin{vmatrix} K_{11} & K_{12} & 0 \\ K_{21} & K_{22} & K_{23} \\ 0 & K_{32} & K_{33} \end{vmatrix} \quad \text{pentru sistemul S} \tag{21}$$

De aceea este necesară introducerea unei componente globale a comenzii $\tilde{u}=\tilde{K}_g\tilde{x}$ pe lingă componenta locală $\tilde{u}=\tilde{K}_1x$ astfel ca legea de comandă : $\tilde{u}=(\tilde{K}_1+\tilde{K}_g)\tilde{x}$ să fie comprimabilă. Deşi această reacţie suplimentară nu este descentralizată în extensia $\tilde{S}$, asigură conducerea descentralizată în sistemul S.

Lucrarea prezintă în final o aplicaţie a principiilor teoretice enunţate pe un model cunoscut în literatură [7].

Cazul interacţiunilor dinamice (cu sau fără întîrzieri) între subsistemele interconectate este abordat în *lucrarea 11.1/A4.* Pe baza vectorilor proprii ai subsistemelor care descriu interacţiunile se găseşte spectrul dorit de valori proprii pentru sistemul compus.

Este cunoscut faptul că utilizarea unor legi de comandă care încorporează şi valorile anterioare ale vectorului de stare (istoricul) conduce la performanţe superioare privind robusteţea şi sensibilitatea la modificarea valorilor nominale ale parametrilor sistemului şi la perturbaţiile exterioare [8]. Acest rezultat poate fi utilizat şi în cazul sistemelor interconectate.

Lucrarea 11.1/A5 tratează rolul strategiilor de memorare în jocurile dinamice, în particular în legătură cu existenţa şi unicitatea unor tipuri variate de echilibru. Unităţile de conducere descentralizată sînt privite ca „jucători" din teoria jocurilor.

2. ESTIMATOARE ŞI REGULATOARE DESCENTRALIZATE

Lucrările grupate în cadrul *secţiunii 11.1/B* propun diferite abordări pentru proiectarea estimatoarelor şi regulatoarelor descentralizate, exploatînd structura fizică a sistemelor complexe (compuse din subsisteme interconectate) sau operînd asupra subsistemelor obţinute prin metodele clasice de descompunere. Proiectarea unui regulator descentralizat, pentru fiecare subsistem, atrage după sine necesitatea estimării (observării) de o manieră descentralizată a stărilor, ieşirilor, intrărilor de interacţiune ale subsistemelor.

O primă abordare este prezentată în *lucrarea 11.1/B1*, pentru cazul sistemelor cu două scări de timp (două dinamici : una lentă, cealaltă rapidă). Prin utilizarea metodelor de agregare (înlocuirea modelului original de ordinul n printr-un model redus, de ordinul n_1), informaţia corespunzătoare celor $n-n_1$ stări suprimate este pierdută. Modelul redus (agregat) poate conduce la un sistem de comandă departe de optim, chiar instabil, metodele de agregare nepermiţînd reintroducerea modurilor suprimate. Acest lucru este posibil, aşa cum s-a arătat în [9], în cazul sistemelor care pot fi reprezentate prin modele singular perturbate, şi anume cu un parametru mic ε acţionînd multiplicativ asupra derivatelor în timp ale variabilelor de stare :

$$\begin{aligned} \dot{x} &= f\,(xyu\,\varepsilon t) \\ \varepsilon\dot{y} &= g\,(xyu\,\varepsilon t) \end{aligned} \tag{1}$$

Astfel de sisteme se caracterizează prin existența a două dinamici, una lentă, luată în considerație la primul pas, cealaltă rapidă, care poate fi tratată separat. Cînd aceste două dinamici pot fi puse în evidență se poate aplica cea mai mare parte a rezultatelor obținute în cazul sistemelor singular perturbate.

Pentru separarea dinamicii globale a sistemului în două părți se folosesc diverse tehnici, dintre care în lucrare este prezentată tehnica de triunghiularizare. Dat fiind sistemul dinamic:

$$\dot{X} = AX \tag{2}$$

partiționat arbitrar sub forma:

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{3}$$

prin utilizarea transformării:

$$\begin{bmatrix} X_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} = \mathfrak{L} \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \tag{4}$$

sistemul poate fi pus sub forma bloc triunghiularizată:

$$\begin{bmatrix} \dot{X}_1 \\ \dot{Z}_2 \end{bmatrix} = \begin{bmatrix} \tilde{A}_1 & \tilde{A}_{12} \\ R(L) & \tilde{A}_2 \end{bmatrix} \begin{bmatrix} X_1 \\ Z_2 \end{bmatrix} \tag{5}$$

unde:

$$\tilde{A}_1 = A_{11} - A_{12}L \quad \text{și} \quad \tilde{A}_2 = A_{22} + LA_{12} \tag{6}$$

L — soluția ecuației Riccati:

$$R(L) = 0 = A_{21} + LA_{11} - LA_{12} - A_{22}L \tag{7}$$

Calculul matricii de transformare L se poate face fie prin utilizarea vectorilor proprii, fie prin utilizarea polinoamelor caracteristice.

O contribuție importantă o aduce lucrarea în domeniul conducerii sistemelor cu două scări de timp. Utilizînd transformarea (4), considerind și matricile de intrare (B) și ieșire (C), sistemul (2) este pus sub forma:

$$\begin{bmatrix} \dot{X}_1 \\ \dot{X}_2 \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} u \tag{8}$$

$$\begin{bmatrix} \dot{X}_1 \\ Z_2 \end{bmatrix} = \begin{bmatrix} A_s & A_{12} \\ 0 & A_f \end{bmatrix} \cdot \begin{bmatrix} X_1 \\ Z_2 \end{bmatrix} + \begin{bmatrix} B_1 \\ B_f \end{bmatrix} u$$

$$y = [C_1 \; C_2] \begin{bmatrix} X_1 \\ X_2 \end{bmatrix} \quad y = [C_s \; C_2] \begin{bmatrix} X_1 \\ Z_2 \end{bmatrix} \tag{9}$$

cu: $A_s = A_{11} - A_{12}L$ $\quad A_f = A_{22} + LA_{12}$

$B_f = LB_1 + B_2$ $\quad C_s = C_1 - C_2L$

Considerind dinamicile A_s și A_f distincte, intrarea u compusă din componente lente u_s și rapide u_f (u_s transmisă direct de $-A_f^{-1} B_f$, iar u_f puternic atenuată de A_s), sistemul este reprezentat ca în fig. 1. Această descompunere permite sinteza utilizînd reacția de la stările și ieșirile decuplate, mai ușor decit utilizînd informația de stare globală. Dacă K_0 și K_f sînt corectorii de stare proiectați pentru cele 2 subsisteme — lent și rapid — atunci este suficientă identificarea lui u:

$$u = K_0X_s + K_fX_f = K_1X_1 + K_2X_2 \tag{10}$$

Conform fig. 1 :

$$X_2 = X_l + X_{sl} - LX_0 = X_l - A_l^{-1} B_l K_0 X_s - LX_s \quad (11)$$

rezultînd :

$$K_1 K_2 = [(1 + K_l A_l^{-1} B_l) K_0, \ K_l] \begin{bmatrix} I & 0 \\ L & I \end{bmatrix} \quad (12)$$

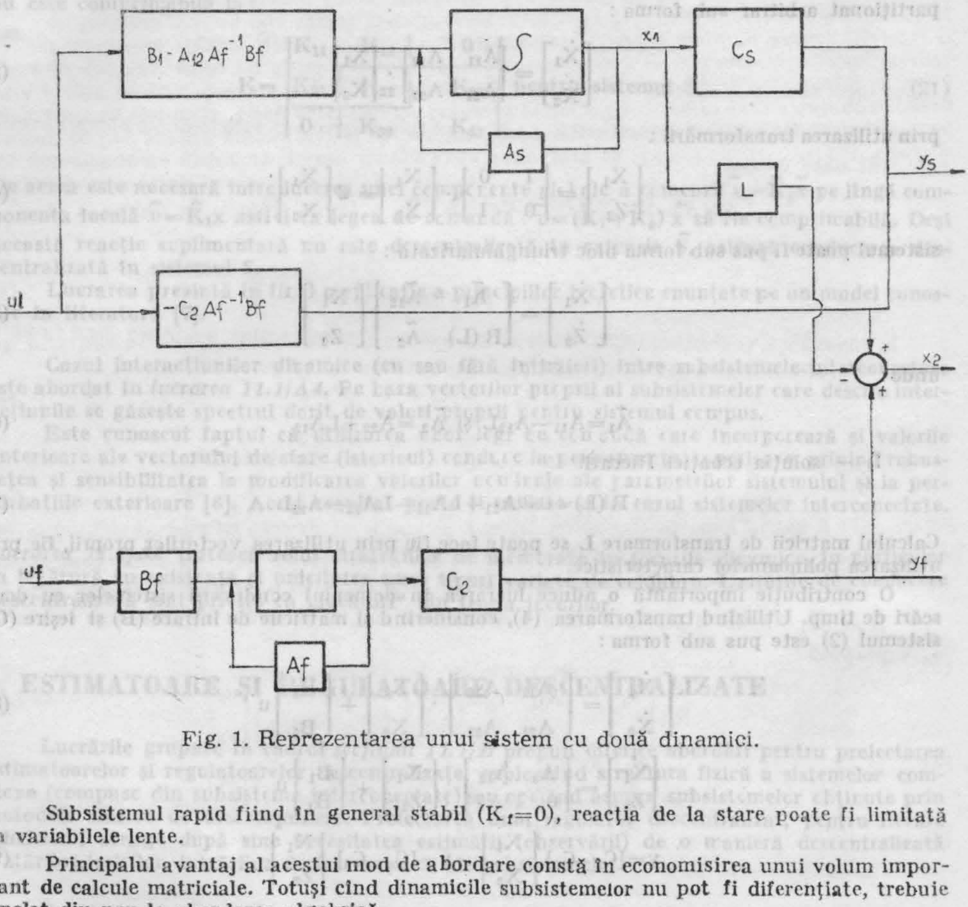


Fig. 1. Reprezentarea unui sistem cu două dinamici.

Subsistemul rapid fiind în general stabil ($K_l = 0$), reacţia de la stare poate fi limitată la variabilele lente.

Principalul avantaj al acestui mod de abordare constă în economisirea unui volum important de calcule matriciale. Totuşi cînd dinamicile subsistemelor nu pot fi diferenţiate, trebuie apelat din nou la abordarea algebrică.

În *lucrarea 11.1/B3* la sinteza estimatoarelor descentralizate este utilizată informaţia privitoare la vectorul local de stare $x_i(t)$ şi intrarea de cuplare (de interacţiune) $h_i(t)$. În sistemul dinamic considerat :

$$z_1(t) = F_1 z_1(t) + G_1 y_1(t) + J_1 w_1(t) \qquad z_1(t_0) = z_{10} \quad (13)$$

$$w_1(t) = L_1^1 z_1(t) + L_1^2 y_1(t)$$

ieşirea $w_i(t)$ trebuie să conveargă către o funcţie necunoscută $p_i(t)=M_i x(t)$ de stările sistemului observat. Aşa cum s-a menţionat, se consideră :

$$p_i(t)=[X_i^T \mid h_i^T]^T \tag{14}$$

$$M_i=\begin{bmatrix} \underline{0} & \dots & \underline{0} & \underline{I} & \underline{0} & \dots & \underline{0} \\ A_{i1} & \dots & A_{ii-1} & \underline{0} & A_{ii+1} & \dots & A_{iN} \end{bmatrix}$$

unde : N este numărul de subsisteme.

În acest caz estimatorul realizează o funcţie lineară de stările locale $x_i(t)$ şi intrările de interacţiune $h_i(t)$ (fig. 2). S-ar putea folosi $p_i(t)=x(t)\,(M_i=\underline{I})$ (maximum de informaţie despre sistemul global), dar efortul de proiectare ar creşte considerabil şi în plus ar fi estimate stări care nu ar intra în componenţa comenzii descentralizate.

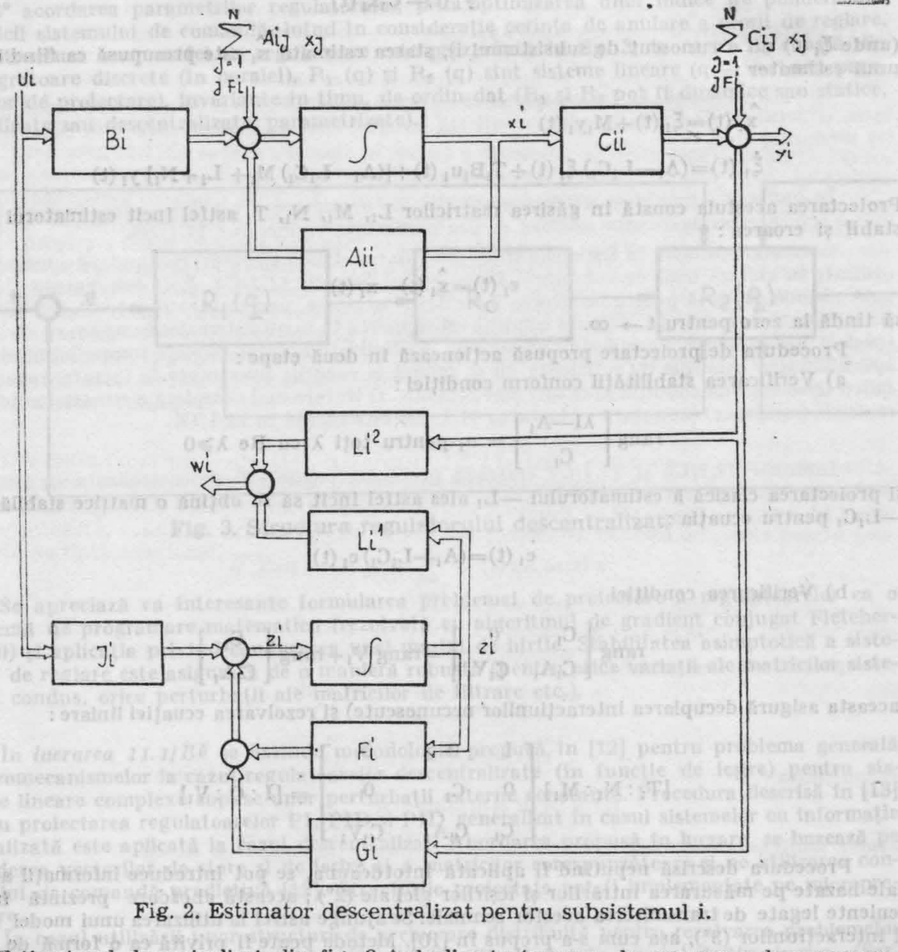


Fig. 2. Estimator descentralizat pentru subsistemul i.

Estimatorul descentralizat din fig. 2 îndeplineşte teorema de separare ; de asemenea ecuaţiile diferenţiale ale erorii de estimare ale subsistemelor sînt decuplate. Prin considerarea ieşirilor estimatorului ca intrări ale subsistemului i : $u_i(t)=w_i(t)$ se aplică estimatorul la conducerea descentralizată, în lucrare prezentîndu-se condiţiile de existenţă ale acestui estimator de comandă şi aplicaţia la un sistem de conducere a unei turbine (8 stări, 2 subsisteme). Alte

aplicaţii posibile sînt sistemele de conducere pentru coloane de distilare şi pentru cuptoare cu încălzire continuă.

În *lucrarea 11.1./B4* sînt prezentate trei structuri de bază pentru estimatoarele de stare descentralizate :

1° estimator complet descentralizat, utilizînd exclusiv date şi semnale ale subsistemului ;

2° estimator parţial descentralizat, utilizînd şi semnale de intrare şi/sau ieşire globale ;

3° estimator descentralizat incluzînd un model de ordin redus al interacţiunilor.

În cazul 1°, considerînd subsistemul S_i :

$$\dot{x}_i(t)=A_i x_i(t)+B_i u_i(t)+V_i \zeta_i(t) \qquad (15)$$

$$y_i(t)=C_i x_i(t)$$

(unde $\zeta_i(t)$ nu e cunoscut de subsistemul i), starea estimată $\hat{x}_i$ este presupusă ca fiind ieşirea unui estimator :

$$\hat{x}_i(t)=\xi_i(t)+M_i y_i(t)$$

$$\dot{\xi}_i(t)=(A_i-L_iC_i)\,\xi_i(t)+T_iB_iu_i(t)+[(A_i-L_iC_i)M_i+L_i+N_i]\,y_i(t) \qquad (16)$$

Proiectarea acestuia constă în găsirea matricilor L_i, M_i, N_i, T_i astfel încît estimatorul să fie stabil şi eroarea :

$$e_i(t)=\hat{x}_i(t)-x_i(t) \qquad (17)$$

să tindă la zero pentru $t\to\infty$.

Procedura de proiectare propusă acţionează în două etape :

a) Verificarea stabilităţii conform condiţiei :

$$\text{rang}\begin{bmatrix}\lambda I-A_i\\ C_i\end{bmatrix}=n_i \text{ pentru toţi } \lambda \text{ cu Re } \lambda\geqslant 0 \qquad (18)$$

şi proiectarea clasică a estimatorului —L_i ales astfel încît să se obţină o matrice stabilă A_i—L_iC_i pentru ecuaţia :

$$\dot{e}_i(t)=(A_i-L_iC_i)\,e_i(t) \qquad (19)$$

b) Verificarea condiţiei :

$$\text{rang}\begin{bmatrix}C_i & 0\\ C_iA_i & C_iV_i\end{bmatrix}=\text{rang}\,V_i+\text{rang}\begin{bmatrix}C_i\\ C_iA_i\end{bmatrix} \qquad (20)$$

(aceasta asigură decuplarea interacţiunilor necunoscute) şi rezolvarea ecuaţiei liniare :

$$[T_i : N_i : M_i]\begin{bmatrix}I & 0 & 0\\ 0 & C_i & 0\\ C_i & C_iA_i & C_iV_i\end{bmatrix}=[I : O : V_i] \qquad (21)$$

Procedura descrisă neputînd fi aplicată întotdeauna, se pot introduce informaţii adiţionale bazate pe măsurarea intrărilor şi ieşirilor globale (2°) ; această abordare prezintă inconveniente legate de transmiterea acestor semnale. Se ajunge astfel la utilizarea unui model redus al interacţiunilor (3°), aşa cum s-a propus în [10]. Metoda poate fi privită ca o formă de agregare de aproximare a sistemului complement sau ca o modelare a interacţiunilor ca perturbaţii acţionînd din exterior asupra subsistemului S_i.

Stabilind teoreme de existenţă pentru tipurile 1°, 2° şi 3° de estimatoare descentralizate, lucrarea lasă deschise unele aspecte privind analiza sensibilităţii şi condiţiile de stabilitate pentru un sistem de comandă în buclă închisă realizat printr-un estimator.

Alte două lucrări — [11.1/B5] şi [11.1/B6] — se referă la problema proiectării regulatoarelor descentralizate pentru cazul sistemelor liniare.

În *lucrarea 11.1/B5*, pentru cazul sistemelor dinamice discrete, lineare, invariante în timp se propune o metodă de proiectare a regulatoarelor descentralizate bazată pe optimizare parametrică. Modul de abordare este similar cu cel propus în [11] pentru cazul sistemelor dinamice continue. Metoda de proiectare este foarte flexibilă şi poate fi aplicată în cazurile în care asupra structurii regulatoarelor se impun restricţii de tipul :

— regulatoare descentralizate ;

— specificare parţială a relaţiilor funcţionale pe care regulatoarele le stabilesc între intrări şi ieşiri.

Procedura de proiectare propusă — bazată pe optimizare parametrică — cuprinde două etape :

1° alegerea structurii regulatorului care satisface restricţiile de informaţie necesară şi complexitate ;

2° acordarea parametrilor regulatorului, prin optimizarea unui indice de ponderare a dinamicii sistemului de comandă, luînd în consideraţie cerinţa de anulare a erorii de reglare.

Structura unor astfel de regulatoare este prezentată în fig. 3, în care : R_0 constă din p integratoare discrete (în paralel), R_1 (q) şi R_2 (q) sînt sisteme lineare (q — vectorul parametrilor de proiectare), invariante în timp, de ordin dat (R_1 şi R_2 pot fi dinamice sau statice, centralizate sau descentralizate, parametrizate).

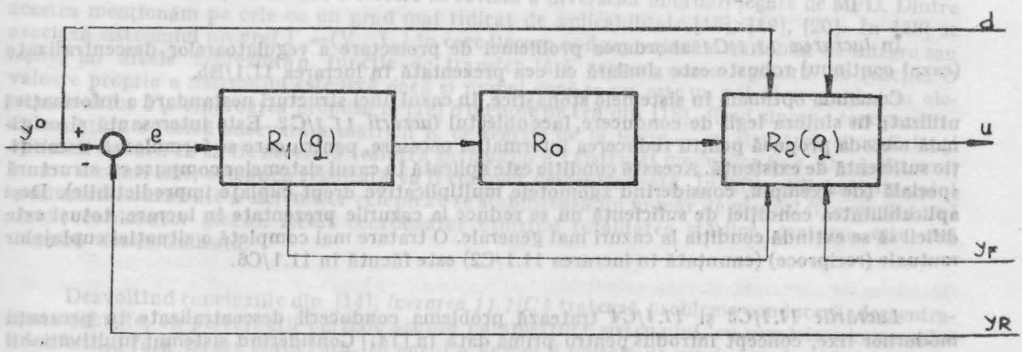


Fig. 3. Structura regulatorului descentralizat.

Se apreciază ca interesante formularea problemei de proiectare a regulatoarelor ca o problemă de programare matematică (rezolvată cu algoritmul de gradient conjugat Fletcher-Powell) şi aplicaţia privind conducerea unei maşini de hîrtie. Stabilitatea asimptotică a sistemului de reglare este asigurată de o manieră robustă (pentru orice variaţii ale matricilor sistemului condus, orice perturbaţii ale matricilor de filtrare etc.).

În *lucrarea 11.1/B6* se extinde metodologia propusă în [12] pentru problema generală a servomecanismelor la cazul regulatoarelor descentralizate (în funcţie de ieşire) pentru sistemele lineare complexe supuse unor perturbaţii externe constante. Procedura descrisă în [13] pentru proiectarea regulatoarelor PI, PID şi PID generalizat în cazul sistemelor cu informaţie centralizată este aplicată la cazul descentralizat. Abordarea propusă în lucrare se bazează pe extinderea vectorilor de stare şi de ieşire şi a matricilor corespunzătoare şi pe utilizarea conceptului de comandă predictivă [13]. Structurile proiectate pot fi implementate pe microprocesoare.

În cazul utilizării unor structuri de prelucrare distribuită pentru rezolvarea problemelor de optimizare şi conducere multinivel, algoritmii utilizaţi sint în general de tip sincron. (Sarcina de prelucrare la o iteraţie este împărţită între procesoarele disponibile, acestea comunicînd informaţiile privind ieşirile iteraţiei curente înainte de începerea iteraţiei următoare). Acest mod de prelucrare urmăreşte ordinea operaţiilor din algoritm şi are avantajul de a simplifica analiza convergenţei. Algoritmii distribuiţi de tip sincron prezintă totuşi dezavantajul

de a necesita un protocol de inițializare a iterației și de a limita viteza de calcul la viteza celui mai lent procesor.

În *lucrarea 11.1/B2* se iau în considerație algoritmi care tolerează o ordine mai flexibilă a calculelor și a comunicațiilor între procesoare (algoritmi asincroni). Astfel de algoritmi au găsit aplicații în rețelele de calculatoare (este citat exemplul rețelei ARPA), unde — în cazul defectării procesoarelor — este complicat să se mențină sincronizările între nodurile întregii rețele cînd acestea execută funcțiile de control în timp real (dirijarea pachetelor de mesaje). Pentru astfel de algoritmi este interesant de stabilit gradul minim de coordonare necesar pentru ca un algoritmi dat să conducă la soluția corectă. Lucrarea se rezumă la enunțarea acestor probleme și a aplicațiilor posibile (optimizare distribuită, estimarea parametrilor, rețelele de calculatoare).

3. CONDUCERE DESCENTRALIZATĂ

Lucrările *secțiunii 11.1/C* completează tematica secțiunii 11.1/B cu aspecte privind proiectarea structurilor de conducere descentralizată în cazul sistemelor stohastice și în cazul prezenței modurilor fixe.

În *lucrarea 11.1/C1* abordarea problemei de proiectare a regulatoarelor descentralizate (cazul continuu) robuste este similară cu cea prezentată în lucrarea 11.1/B5.

Comanda optimală în sistemele stohastice, în cazul unei structuri nestandard a informației utilizate în sinteza legii de conducere, face obiectul *lucrării 11.1/C2.* Este interesantă și originală metoda propusă pentru reducerea informației necesare, pentru care se formulează o condiție suficientă de existență. Această condiție este aplicată în cazul sistemelor compuse cu structură specială (de exemplu, considerînd zgomotele multiplicative drept cuplaje impredictibile). Deși aplicabilitatea condiției de suficiență nu se reduce la cazurile prezentate în lucrare, totuși este dificil să se extindă condiția la cazuri mai generale. O tratare mai completă a situației cuplajelor mutuale (reciproce) (enunțată în lucrarea 11.1/C2) este făcută în 11.1/C6.

Lucrările 11.1/C3 și *11.1/C4* tratează problema conducerii descentralizate în prezența modurilor fixe, concept introdus pentru prima dată în [14.] Considerînd sistemul multivariabil linear invariant în timp:

$$\dot{x}(t) = Ax(t) + \sum_{i=1}^{n} B_i u_i(t) \qquad i = \overline{1, n} \tag{1}$$

$$y_i = C_i x(t)$$

cu regulatoarele descentralizate:

$$u_i(t) = K_i y_i(t) + Q_i z_i(t) \tag{2}$$

$$\dot{z}_i(t) = S_i z_i(t) + R_i y_i(t)$$

și mulțimea:

$$K^* = \{K / K = \text{blocdiag}[K_1, \ldots, K_n]\} \tag{3}$$

atunci mulțimea modurilor fixe descentralizate (MFD) în raport cu K este:

$$\Lambda(C, A, B, K) = \bigcap_{K \in K^*} \sigma(A + BKC) \tag{4}$$

în care: $(\cdot)$ reprezintă mulțimea valorilor proprii pentru $(\cdot)$;

$$B = B_1, B_2, \ldots, B_n \tag{5}$$

$$C^T = [C_1^T, C_2^T, \ldots, C_n^T]^T$$

Condiția necesară și suficientă de existență a regulatoarelor descentralizate (2) pentru ca sistemul (1) să fie asimptotic stabil este ca MFD ale lui (1) să fie toate în partea stîngă deschisă a planului complex. MFD joacă un rol important în multe probleme de conducere (problema stabilizării descentralizate, problema servomecanismului descentralizat robust) și în determinarea controlabilității și observabilității în sistemele puternic interconectate. O descriere algebrică a MFD a fost realizată în [15]. În [16] s-a introdus conceptul de moduri fixe structurate. În cazul cînd MFD sînt o reflectare a parametrilor sistemului, acestea pot fi eliminate prin modificarea parametrilor ; cînd MFD reflectă structura sistemului, eliminarea lor se poate face fie modificînd structura sistemului fie relaxînd restricțiile impuse asupra fluxului de informații între regulatoarele locale (schimbarea structurii matricii de reacție K). MFD pot fi eliminate printr-o descompunere adecvată a sistemului (cînd aceasta nu este impusă). Cînd descompunerea sistemului este impusă și are MFD, pentru eliminarea acestora se poate recurge la procedura propusă în [17] ; MFD sînt caracterizate prin matrici dominante blocdiagonale și se determină mulțimea de legături între subsisteme (blocuri K_{ij}) care elimină MFD (pentru matricea K rezultă o structură redundantă).

În lucrarea 11.1/C3 se propune o nouă modalitate de caracterizare a MFD. Sînt deduse condițiile necesare și suficiente pentru ca un pol al sistemului să fie un MFD și unele condiții suficiente pentru ca un sistem să nu aibă MFD. Cînd se obțin MFD instabile este sugerată modalitatea de alegere a unei noi structuri a informației de control, care elimină MFD (cu condiția ca sistemul să fie controlabil și observabil). Pentru exemplificare se folosesc exemplele cunoscute în literatura de specialitate [17].

În lucrarea 11.1/C4 se face o trecere în revistă a diverselor abordări legate de MFD. Dintre acestea menționăm pe cele cu un grad mai ridicat de aplicabilitate [18], [19], [20]. În [18] se asociază sistemului un graf $\Gamma_s=(V_s, L_s)$ în care fiecare nod reprezintă o variabilă de intrare sau ieșire, iar arcele reprezintă o funcție de transfer fără zerouri sau o legătură de reacție. O valoare proprie a sistemului este fixă dacă și numai dacă ea nu este un pol al vreunui ciclu elementar al lui Γ_s. Abordarea propusă în [19] o include pe cea din [18], fiind singura care permite să se obțină o formă adecvată a matricii K de reacție de la ieșire — cu structură și coeficienți optimali — fără ca să fie necesară testarea prealabilă a existenței modurilor fixe. Dacă sistemul nu are moduri fixe, comanda rezultată va fi complet descentralizată. În [20], utilizînd principiul conducerii ierarhizate a sistemelor complexe se stabilizează un număr de moduri fixe cu ajutorul unui coordonator global. Acest coordonator utilizînd totalitatea stărilor, tratarea este mai degrabă descentralizată.

Dezvoltînd concluziile din [14], *lucrarea 11.1/C5* tratează problema conducerii descentralizate robuste, cu perturbații parțiale asupra parametrilor sistemului, cu urmărirea comenzilor de referință fără eroare staționară. În cazul a numai 2 unități de conducere locală și în ipotezele menționate, condițiile necesare și suficiente din [14] devin condiții suficiente pentru problemele de tipul menționat.

În *lucrarea 11.1/C6*, pe baza lucrărilor [21], [22], se propune o trunchiere a interconexiunilor considerîndu-se pentru ele o formă simetrică (mutuală), ceea ce apare ca o presupunere mai realistă decît ignorarea lor totală. Pentru sistemul stohastic :

$$dx=[A_D\, dt+A_C\,(dw)]\, x+Bu\, dt \tag{6}$$

$$A_C=(a_{ij}) \quad i, j=\overline{1, n}$$

cu indicele de performanță :

$$J=\int_0^{\infty} [x^T(t)\, Qx(t)+u^T(t)\, Ru(t)]\, dt \tag{7}$$

descompus în s subsisteme :

$$dx_i=A_i x_i dt+\sum_{j=1}^{s} A_{ij}\,(dw)\, x_j+B_i u_i dt \quad i=\overline{1, s} \tag{8}$$

$$J_i=\int_0^{\infty} [x_i^T(t)\, a_i x_i(t)+u_i^T(t)\, R_i u_i(t)]\, dt, \tag{9}$$

problema conducerii descentralizate optimale constă în calculul legii de comandă cu restricţii:

$$u^T(x)=[u_1(x_1), u_2(x_2), \ldots, u_s(x_s)] \quad (10)$$

care minimizează valoarea medie a indicelui J.

Se impun condiţiile:

$$1° \quad E(a_l a_j^T)=-E(a_l a_j^T)^T \gamma, \quad \nu=1, s \quad \gamma\neq\nu \quad (11)$$

(Acţiunile componentelor x^l şi x^j asupra componentelor x^k şi x^l sînt opuse şi linear dependente — în valoare medie).

Introducînd constanta:

$$\beta=\inf_K \left\| \int_0^\infty \exp[A_D-BK)^T t] \exp[(A_D-BK)t]\, dt \right\| \quad (12)$$

în care $\|D\|$ — norma vectorială euclidiană, condiţia:

$$2° \quad \sum_{k=1}^{P} \|G_k\|^2 \leqslant (\beta P)^{-1} \quad (13)$$

semnifică exercitarea unor interacţiuni slabe între subsisteme.

Considerînd sistemul stohastic interconectat (6), condiţiile 1° şi 2° îndeplinite şi (A_D, B) controlabile, legea de comandă:

$$u^\circ(x)=-Kx \quad (14)$$

satisface (10) şi minimizează indicele de performanţă:

$$EJ=E\int_0^\infty (x^T Qx+u^T Ru)\, dt \quad (15)$$

Matricea de comandă K este calculată cu:

$$K=\mathrm{diag}[K_1, K_2, \ldots, K_s] \quad (16)$$

în care: $K_k=P_k^{-1}B_k^T P_k$

iar P_k ($k=\overline{1, s}$) sînt blocuri ale diagonalei matricii pozitiv definite: $P=\mathrm{diag}[P_1, P_2, \ldots, P_s]$ care satisfac sistemul de ecuaţii algebrice:

$$A_k^T P_k+P_k A_k-P_k B_k R_k^{-1} B_k^T P_k+Q_k+\Gamma_h(P)=0$$

în care:

$$\Gamma_k(P)\, dt=E(a_i^T P a_j) \quad k=\overline{1, s} \quad (17)$$

Matricea A_D—BK este stabilă. Mai mult, sistemul stohastic în buclă închisă (6) cu legea de comandă (14) este (cu probabilitatea 1) exponenţial stabil şi ordinul de stabilitate cel mai mic este:

$$\alpha=\lambda_m(Q+K^T RK)/\lambda_M(P) \quad (18)$$

în care λ_m şi λ_M sînt valorile proprii minime şi maxime ale matricilor respective.

Creşterea volumului de calcule datorită cuplajelor ecuaţiilor Riccati este compensată de faptul că procedura furnizează imediat o structură de conducere descentralizată stabilă.

În concluzie lucrările acestei secţiuni dezvoltă principiile conducerii descentralizate a sistemelor interconectate stabilite în [14], [21] şi [22].

BIBLIOGRAFIE

1. *Kolmogorov*, A.N. (1965). **Three aproaches for defining the concept of information quantity.** Prob. Inform. Trans. 1, p. 1–7.
2. *Chaitin, G.J.* (1966). **On the length of programs for computing finite binary sequences.** Journal of ACM, 13, p. 547–569.
3. *Liu Yong-Qing* (1965). * **Decomposition of Liapunov's function.** ACTA Automatica Sinica vol. 3, No. 3.
4. *Michel, A.N., Miller, R.K.* (1977). **Qualitative analysis of large-scale dynamical system.** Academic Press.
5. *Siljak, D.D.* (1978). **Large-scale dynamic systems, stability and structure.** North Holland.
6. *Bailey, F.N.* (1966). **The application of Lyapunov's second method to interconnected systems.** J. SIAM Control Ser. A.3, p. 443–462.
7. *Levine, W.S., Athans, M.* (1966). **On the optimal error regulation of a string of moving vehicles.** IEEE Transactions on Automatic Control, AC-11, p. 355–361.
8. *Sundarashan, N., Cruz, J.B.* (1972). **Sensitivity reduction in timevarying linear and nonlinear systems.** Int. Journal of Control, 15, 5, p. 937–943.
9. *Kokotovic P.V., Yackel R.A.* (1977) **Singular perturbation of linear regulators : basic theorems.** IEEE Trans. on Autom. Control — Vol. AC 17, no. 1.
10. *Tacker, E.C., Sanders, C.W.* (1982). **Decentralized structures for state estimation in large scale systems.** Large Scale Systems 3, p. 255–266.
11. *Guardabassi, G., Locatelli, A., Maffezzoni, C., Schiavoni, N.* (1979). **Parameter optimization in decentralized process control : a unified setting for multivariable industrial regulator design.** M.A. Cuénod (Ed.), Proc. IFAC Symposium on Computer Aided Design of Control Systems, Pergamon-Press, Oxford, p. 87–92.
12. *Davison, E.J.* (1976). **The robust Decentralized Control of General Servomechanism Problem.** IEEE Trans. on Automatic Control, 21, p. 14–24.
13. *Medanic, J.* (1979). **Synthesis of Decentralized Output Regulators.** Systems Engineering for Power Vol. II, DOE, Davos Switzerland, p. 3 121–3 134.
14. *Davison, E.H., Wang, S.H.* (1973). **On the stabilization of decentralized control systems.** IEEE Trans. on Automatic Control, vol. AC-18, p. 473–478.
15. *Anderson, B.D.O., Clements, D.J.* (1981). **Algebraic characterization of fixed modes in decentralized control.** Automatica, 17, p. 703–712.
16. *Sezer, M.E., Siljak, D.D.* (1981). **Structurally fixed modes.** Systems and control letters, vol. 1, no. 1, p. 60–64.
17. *Armentano, V.A., Singh, M.G.* (1981). **A new approach to the decentralized controller initialization problem.** Preprints IFAC Congress, Kyoto.
18. *Locatelli, A., Schiavoni, N., Tarantini, A.* (1977). **Pole placement : role and choice of the underlying information pattern.** Richerche di Automatica, vol. 18, no. 1., p. 107–126.
19. *Senning, M.F.* (1979). **Feasibly decentralized control.** Ph. D. Thesis ETH Zürich.
20. *Groumpos, P.P., Loparo* (1980). **Structural control of large-scale systems.** 19-th IEEE Conference on Decision and Control.
21. *Sandell, N.R., Varaiya, P., Athans, M., Safonov, M.G.* (1978). **Survey of Decentralized Control Methods for Large Scale Systems.** IEEE Transactions, AC-23, p. 108–128.
22. *Wonham, N.M.* (1967). **Optimal Stationary Control of a Linear System with State — Dependent Noise.** SIAM Journal of Control, 5, p. 486–500.

4. APLICAȚII ALE METODOLOGIILOR SISTEMELOR COMPLEXE

După cum s-a arătat într-o lucrare anterioară [1] dedicată sintezei unor comunicări științifice legate de sistemele mari și complexe prezentate la al 8-lea Congres IFAC de la Kyoto, eforturi importante de cercetare sînt depuse în vederea rezolvării problemelor integrate de optimizare și estimare a parametrilor în cazul dinamic. Și în cadrul grupajului de analizat în prezenta

sinteză, două lucrări au abordat problema amintită. Astfel Singh (11.1/D6], în contextul unei aplicaţii legate de procesele de fermentaţie, arată că procesul poate fi reprezentat de modelul matematic:

$$\underline{\dot{x}}=\underline{f}(\underline{x}, \underline{m}, \underline{\alpha}) \quad (1)$$

unde: vectorul de stare, $\underline{x}$, indică creşterea celulelor şi sinteza penicilinei, vectorul de comandă $\underline{m}$ reprezintă temperaturile, iar vectorul $\underline{\alpha}$ conţine coeficienţii de reacţie catalitică enzimatică.

Funcţia obiectiv de extremizat este de tipul Mayer:

$$J=\underline{c}^{T}\underline{x}(t_f) \quad (2)$$

unde $\underline{c}$ indică un vector constant, t_f este timpul final, iar T indică transpunerea.

În condiţiile în care $\underline{\alpha}$ nu este cunoscut cu precizie, problemele de optimizare şi estimare a parametrilor se pot combina într-o problemă de extremizare (minimizare) a următorului criteriu combinat:

$$J_1=\min_{n}\left\{\frac{1}{2}(1-\beta)\int_0^T(\underline{x}^T Q\underline{x}+\underline{m}^T R\underline{m})\,dt+\frac{1}{2}\beta\int_0^T(y^\circ-y)^T W(y^\circ-y)\,dt\right\} \quad (3)$$

cu restricţiile (1) şi

$$\underline{y}=\underline{g}(\underline{x}, \underline{m}, \underline{\alpha}) \quad (4)$$

unde y° este vectorul mărimilor măsurate, iar β este un coeficient subunitar pozitiv.

Procedura constă în rezolvarea iterativă a problemei definite de criteriul (3) şi restricţiile (1) şi (4) şi aplicarea comenzilor $\underline{m}$ procesului real în condiţiile scăderii continue a valorii parametrului β. În lucrare se arată că procedura a fost implementată pe un microcalculator.

Dacă elementele prezentate mai sus se refereau la o problemă dinamică, pentru cazul static, Ellis, Mihalska şi Roberts (11.1/D3). prezintă extinderi ale unor rezultate proprii anterioare [2] în cazul sistemelor interconectate. Dată fiind importanţa metodei, care permite, în principiu, economii substanţiale atît în efortul de calcul (deoarece admite folosirea unor modele simple), cît şi, uneori, în costul instrumentaţiei de măsurare, în condiţiile obţinerii de valori concrete optimale ale referinţelor regulatoarelor pentru procese complicate, în continuare va fi descrisă pe larg.

Fie un sistem format din N subsisteme interconectate, prevăzute cu bucle de reglare, a căror funcţionare în regim static este dată de:

$$\underline{y}_i=\overset{*}{f}_i(\underline{c}_i, \underline{u}_i) \qquad i=\overline{1, N} \quad (5)$$

unde y_i, c_i şi u_i sînt ieşirile, consemnele regulatoarelor şi respectiv intrările de interconexiune ale subsistemelor aparţinînd mulţimilor $\mathcal{Y}_i$, $\mathcal{C}_i$ şi $\mathcal{U}_i$.

Cuplajul dintre subsisteme este dat de:

$$\underline{u}_i=\sum_{j=1}^{N} H_{ij}y_j, \quad i=\overline{1, N} \quad (6)$$

unde elementele matricii de interconexiune, H, au valoarea zero sau unu.

Descrierea globală a sistemului este:

$$\underline{y}=\overset{*}{\underline{f}}(\underline{c}, \underline{u}) \quad (5')$$

unde vectorii globali (la nivelul sistemului) sint :

$$\underline{y} \stackrel{\Delta}{=} [\underline{y}_1^T \dots \underline{y}_N^T]^T \in \mathcal{Y}_1 \times \dots \times \mathcal{Y}_N \stackrel{\Delta}{=} \mathcal{Y}$$

$$\underline{u} \stackrel{\Delta}{=} [\underline{u}_1^T \dots \underline{u}_N^T]^T \in \mathcal{U}_1 \times \dots \times \mathcal{U}_N \stackrel{\Delta}{=} \mathcal{U}$$

$$\underline{c} \stackrel{\Delta}{=} [\underline{c}_1^T \dots \underline{c}_N^T]^T \in \mathcal{C}_1 \times \dots \times \mathcal{C}_N \stackrel{\Delta}{=} \mathcal{C}$$

Deoarece în cazul unui sistem complex funcţiile corecte ale subsistemelor (1) pot să nu fie cunoscute cu precizie, sau să fie prea complicate pentru o folosire on-line, se pot folosi modele simplificate :

$$\underline{y}_i = f_i(\underline{c}_i, \underline{u}_i, \underline{\alpha}_i), \text{ sau} \tag{7}$$

$$\underline{y} = f(\underline{c}, \underline{u}, \underline{\alpha})$$

unde : $\underline{f} = [\underline{f}_1^T \dots \underline{f}_N^T]^T$, $\underline{\alpha}_i$ reprezintă parametri modelelor.

Se presupune (cu suficientă acoperire practică) că funcţia de cuplaj adevărată (6) poate fi cunoscută cu precizie. De asemenea, funcţiile care definesc restricţia $\underline{c} \in \mathcal{C}$ pot fi definite cu precizie (în continuare se vor considera numai restricţiile privind variaţia valorilor consemnelor).

Ţinînd seama de structura interconectată a sistemului, într-o problemă de optimizare apare ca naturală considerarea unui criteriu de performanţă aditiv de tipul

$$\min_{c \in \mathcal{C}} \left\{ J(\underline{c}, \underline{u}, \underline{y}) = \sum_{i=1}^{N} J_i(\underline{c}_i, \underline{u}_i, \underline{y}_i) \right\} \tag{8}$$

În definirea problemei de optimizare, pe lîngă criteriul de performanţă (8), se vor considera restricţiile de model (7).

Să considerăm cazul particular cînd în definirea criteriului de performanţă (8) nu intervin şi intrările de interacţiune u_i :

$$J(\underline{c}, \underline{y}) = \sum_{i=1} J_i(\underline{c}_i, \underline{y}_i)$$

Modelul subprocesului (7) poate fi rescris în forma :

$$\underline{y}_i = \underline{f}_i(\underline{c}_i, \underline{\alpha}_i) = \underline{\bar{f}}_i(\underline{c}_i) + \underline{\alpha}_i ; \quad i = \overline{1, N} \tag{9}$$

Rezultă

$$\min_{c \in \mathcal{C}} \{ J(\underline{c}, \underline{y}) = J(\underline{c}, \underline{y}(\underline{c}, \underline{\alpha})) \} \tag{10}$$

Presupunînd că toate variabilele de ieşire $\underline{y}$ care intervin în formularea criteriului de performanţă (10) sînt măsurabile, problema de estimare a vectorului parametrilor i este :

$$f_i(\underline{c}_i \underline{\alpha}_i) = \overset{*}{\underline{y}}_i \rightarrow \underline{\alpha}_i ; \quad i = \overline{1, N} \tag{11}$$

Soluţia acestei probleme este imediată :

$$\underline{\alpha}_i = \overset{*}{\underline{y}}_i - \underline{\bar{f}}_i(\underline{c}_i) ; \quad i = \overline{1, N}$$

unde $\overset{*}{\underline{y}}_i$ este valoarea măsurată a vectorului de ieşire al subsistemului i.

Datorită diferenţelor posibile între model şi realitate, în mod inevitabil, apar interacţiuni între problema de optimizare definită de (10) şi cea de estimare a parametrilor definită de (11).

Pentru a ocoli aceste interacţiuni şi a decupla cele două probleme, vectorul $\underline{\alpha} \triangleq [\underline{\alpha}_1^T \cdots \underline{\alpha}_N^T]^T$ este înlocuit prin vectorul auxiliar $\underline{\sigma}=[\underline{\sigma}_1^T \ldots \underline{\sigma}_N^T]^T$ în probleme de optimizare, iar vectorul $\underline{c}$ este înlocuit cu vectorul auxiliar $\underline{v}=[\underline{v}_1^T \ldots \underline{v}_N^T]^T$ în problema de estimare a parametrilor. Ca urmare, apar următoarele restricţii egalitate suplimentare:

$$\underline{v}=\underline{c} \tag{12}$$

$$\underline{\sigma}=\underline{\alpha} \tag{13}$$

Din (10) şi (12) se redefineşte problema de optimizare:

$$\min_{\underline{c} \in \mathcal{C}} J(\underline{c}, \underline{y}(\underline{c}, \underline{\sigma})) \tag{14}$$

Ţinînd seama de (11) şi (13) problema de estimare a parametrilor devine:

$$\overset{*}{\underline{y}}_i=\underline{f}_i(\underline{v}_i, \underline{\alpha}_i) \tag{15}$$

Pentru rezolvarea problemei integrate de optimizare şi estimare a parametrilor, definită de (14), (15) şi restricţiile (12) şi (13), se poate defini Lagrangeanul:

$$\mathcal{L}=J(\underline{c}, \underline{f}(\underline{c}, \underline{\sigma}))+\underline{\lambda}^T(\underline{v}-\underline{c})+\underline{\mu}^T(\underline{\sigma}-\underline{\alpha})+\underline{\eta}^T(\underline{f}(\underline{v}, \underline{\alpha})-\overset{*}{\underline{y}}) \tag{16}$$

unde $\underline{\lambda}, \underline{\mu}, \underline{\eta}$ reprezintă multiplicatorii Lagrange.

Presupunînd că derivatele diferitelor funcţii există şi sînt continue, din aplicarea condiţiilor de staţionaritate, rezultă problema de optimizare modificată:

$$\min_{\underline{c} \in \mathcal{C}} \{J(\underline{c}, \underline{f}(\underline{c}, \underline{\sigma}))-\underline{\lambda}^T\underline{c}\} \tag{17}$$

unde, tot din condiţiile de staţionaritate, vectorul $\underline{\lambda}$ se calculează cu:

$$\underline{\lambda}=\left(\frac{\partial f^T}{\partial v}-\frac{\partial \overset{*}{y}{}^T}{\partial v}\right)\left(\frac{\partial f^T}{\partial \alpha}\right)^{-1}\left(\frac{\partial J}{\partial \sigma}\right) \tag{18}$$

Ţinînd cont de forma particulară a problemei modificate de optimizare, este o posibilă rezolvare descentralizată:

$$\min_{\underline{c}_i \in \mathcal{C}_i} \{J_i(\underline{c}_i \underline{\sigma}_i)-\underline{\lambda}_i^T \underline{c}_i\} \tag{19}$$

La rîndul său problema de estimare a parametrilor se rezolvă ca şi mai sus, direct, în mod descentralizat:

$$\underline{\alpha}_i=\overset{*}{\underline{y}}_i-\underline{f}_i(\underline{v}_i) \tag{20}$$

Rolul coordonatorului constă în calculul vectorului folosind (18) şi repartizarea valorilor $\underline{\lambda}_i$ către subsistemele locale.

Procedura iterativă de rezolvare on-line a problemei integrate de optimizare şi estimare a parametrilor este ilustrată în fig. 1. La fiecare pas, k, se estimează local vectorii $\underline{\alpha}_i$; $i=\overline{1, N}$, în urma măsurării ieşirilor rezultate obţinute la stabilizarea sistemului după aplicarea referinţelor $\underline{v}_i$. La nivelul coordonatorului se calculează vectorul $\underline{\lambda}$.

Pe baza transferului semnalelor de coordonare $\underline{\lambda}_i$, la nivelul subsistemelor, se rezolvă problemele modificate de optimizare definite de (20).

Pentru a nu perturba prea mult procesul prin modificări substanţiale ale referinţelor, nu se aplică direct soluţiile problemei de optimizare modificate, $\underline{c}$, ci valorile filtrate:

$$\underline{v}_i^{[k]}=\underline{v}_i^{[k-1]}+K_i^{[k]}(\underline{c}_i^{[k]}-\underline{v}_i^{[k-1]}) \tag{21}$$

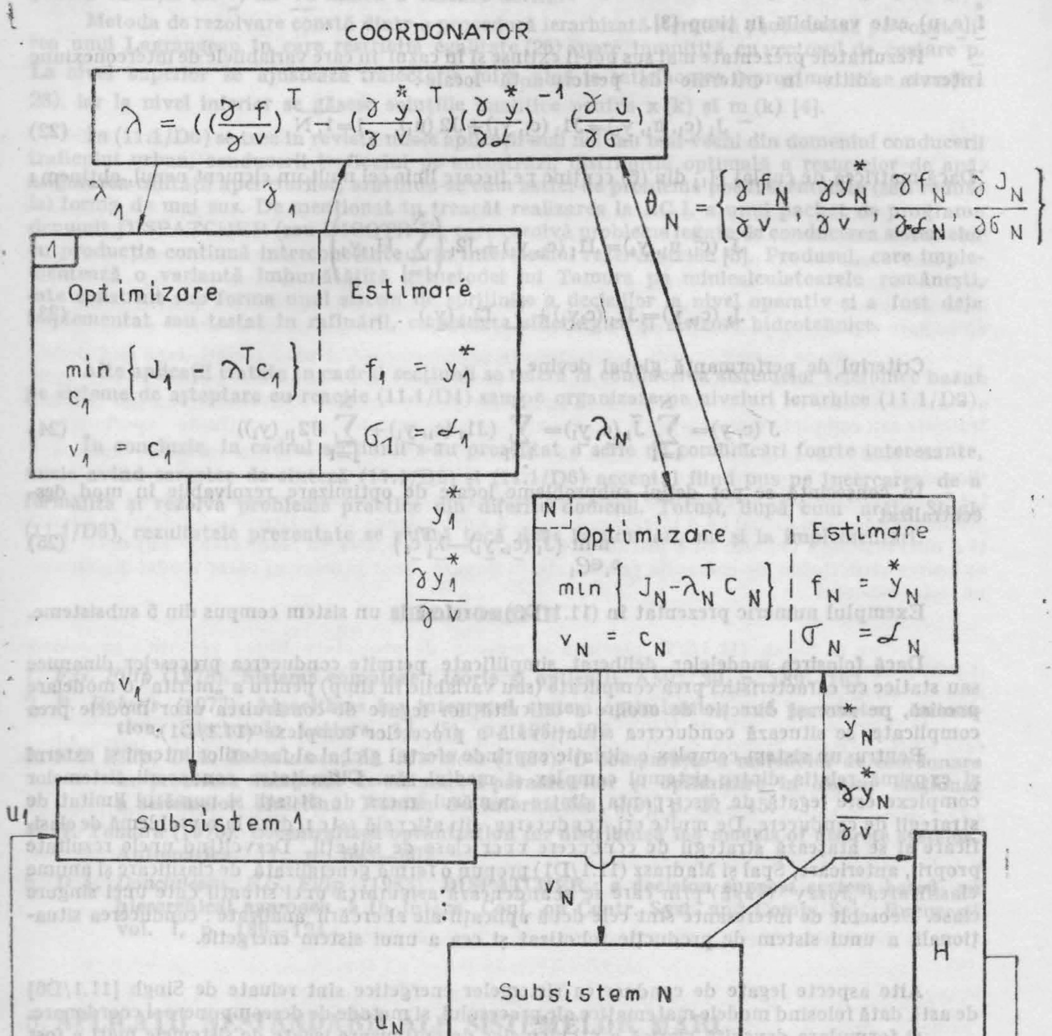


Fig. 4. Rezolvarea on-line a problemei integrate de optimizare şi estimare a parametrilor.

unde k este numărul iterației, iar K este o matrice diagonală cu coeficienți pozitivi subunitari. Vom observa în treacăt că folosirea unei matrici K variabile, cu coeficienți adaptivi poate asigura proprietăți sporite de convergență ale procedurii chiar și atunci cînd funcția procesului $\underline{f}^*(\underline{c}, \underline{u})$ este variabilă în timp [3].

Rezultatele prezentate mai sus pot fi extinse și în cazul în care variabilele de interconexiune intervin aditiv în criteriile de performanță locale:

$$J_i(\underline{c}_i, \underline{u}_i, \underline{y}_i) = J1_i(\underline{c}_i, \underline{y}_i) + J2(\underline{u}_i), \quad i = \overline{1, N} \tag{22}$$

Dacă matricea de cuplaj H_{ij} din (6) conține pe fiecare linie cel mult un element nenul, obținem:

$$J_i(\underline{c}_i, \underline{u}_i, \underline{y}_i) = J1_i(\underline{c}_i, \underline{y}_i) + J2_i\left(\sum_{j=1}^{N} H_{ij}\underline{y}_j\right)$$

$$J_i(\underline{c}_i, \underline{y}) = J1_i(\underline{c}_i\underline{y}_i) + \sum_j J2_{ij}(\underline{y}_j) \tag{23}$$

Criteriul de performanță global devine

$$J(\underline{c}, \underline{y}) = \sum_{i=1}^{N} \tilde{J}_i(\underline{c}_i\,\underline{y}_i) = \sum_{i=1}^{N} (J1_i(\underline{c}_i, \underline{y}_i) + \sum_{j=1}^{N} J2_{ji}(\underline{y}_i)) \tag{24}$$

În consecință se pot defini subprobleme locale de optimizare rezolvabile în mod descentralizat:

$$\min_{\underline{c}_i \in \mathcal{C}_i} \{\tilde{J}_i(\underline{c}_i, \underline{y}_i) - \underline{\lambda}_i^T \underline{c}_i\} \tag{25}$$

Exemplul numeric prezentat în (11.1/D3) se referă la un sistem compus din 5 subsisteme.

Dacă folosirea modelelor deliberat simplificate permite conducerea proceselor dinamice sau statice cu caracteristici prea complicate (sau variabile în timp) pentru a „merita" o modelare precisă, pe aceeași direcție de ocolire a dificultăților legate de construirea unor modele prea complicate, se situează conducerea situațională a proceselor complexe (11.1/D1).

Pentru un sistem complex o situație cuprinde efectul global al factorilor interni și externi și exprimă relația dintre sistemul complex și mediul său. Dificultatea conducerii sistemelor complexe este legată de discrepanța dintre numărul imens de situații și numărul limitat de strategii de conducere. De multe ori, conducerea situațională este redusă la o problemă de clasificare și se atașează strategii de conducere unor clase de situații. Dezvoltînd unele rezultate proprii, anterioare, Spal și Madrasz (11.1/D1) propun o formă generalizată de clasificare și anume clasificarea „fuzzy" (vagă) prin care se abandonează asigurarea unei situații date unei singure clase. Deosebit de interesante sînt cele două aplicații ale abordării analizate: conducerea situațională a unui sistem de producție robotizat și cea a unui sistem energetic.

Alte aspecte legate de conducerea sistemelor energetice sînt reluate de Singh [11.1/D6] de astă dată folosind modele matematice ale procesului. și metode de descompunere și coordonare.

O formulare devenită clasică a problemelor de optimizare legate de sistemele mari a fost propusă de Tamura [4] și este cea descrisă de modelul, restricțiile și criteriul de performanță de mai jos:

$$\underline{x}(k+1) = A\underline{x}(k) + \sum_{j=0}^{m} B_j\underline{m}(k-j) + \underline{w}(k) \tag{26}$$

$$\underline{v}_m \leqslant \underline{v} \leqslant \underline{v}_M \quad \underline{v} = [\underline{x}^T, \underline{m}^T]^T \tag{27}$$

$$J = \sum_{k=0}^{K-1} (\underline{v}(k) - \underline{\underline{vd}}(k))^T R\, (\underline{v}(k) - \underline{\underline{vd}}(k)) \tag{28}$$

unde $\underline{w}$ este vectorul perturbaţiilor, v_m şi v_M arată limitele inferioare şi respectiv superioare pentru variaţia lui $\underline{v}$, iar $\underline{vd}$ indică o valoare dorită.

Metoda de rezolvare constă dintr-o procedură ierarhizată iterativă ; se bazează pe construirea unui Lagrangean în care restricţia egalitate (26) apare înmulţită cu vectorul de costare $\underline{p}$. La nivel superior se ajustează traiectoria lui $\underline{p}$ pînă la satisfacerea (aproximativă a ecuaţiei 28), iar la nivel inferior se găsesc soluţiile analitice pentru $\underline{x}(k)$ şi $\underline{m}(k)$ [4].

În (11.1/D6) se trec în revistă unele aplicaţii mai noi sau mai vechi din domeniul conducerii traficului urban, conducerii traficului pe autostrăzi, distribuţia optimală a resurselor de apă, asigurarea calităţii apei rîurilor, arătîndu-se cum astfel de probleme pot fi formulate (sau reduse la) forma de mai sus. De menţionat în treacăt realizarea la I.C.I. a unui pachet de programe denumit DISPATCHER (sau DICOTR-C), care rezolvă probleme legate de conducerea sistemelor cu producţie continuă interconectate prin intermediul rezervoarelor [5]. Produsul, care implementează o variantă îmbunătăţită a metodei lui Tamura pe minicalculatoarele româneşti, este construit sub forma unui sistem de sprijinire a deciziilor la nivel operativ şi a fost deja implementat sau testat în rafinării, combinate siderurgice şi sisteme hidrotehnice.

Alte aplicaţii tratate în cadrul secţiunii se referă la conducerea sistemelor telefonice bazat pe sisteme de aşteptare cu reacţie (11.1/D4) sau pe organizare pe niveluri ierarhice (11.1/D2).

În concluzie, în cadrul secţiunii s-au prezentat o serie de comunicări foarte interesante, unele avînd caracter de sinteză (11.1/D2) şi (11.1/D6) accentul fiind pus pe încercarea de a formaliza şi rezolva probleme practice din diferite domenii. Totuşi, după cum arăta Singh (11.1/D6), rezultatele prezentate se referă încă doar la simulări, nu şi la implementări.

BIBLIOGRAFIE

1. *F.G. Filip* (1978). **Sisteme complexe : teorie şi aplicaţii,** AMC, 39, p. 189–193.
2. *P. Roberts* (1977). **Algorithms for integrated system optimization and parameter estimation.** Electronic Letters, 14 (5), p. 196–197.
3. *F.G. Filip, D.A. Donciulescu, M. Muraicea* (1984). **O comparaţie a metodelor de coordonare în problema integrată de estimare a parametrilor şi optimizare în regim staţionar a sistemelor.** Buletinul Român de Informatică V(3), p. 31–45.
4. *H. Tamura* (1975). **Decentralized optimization for distributed lag models of discrete systems.** Automatica, 11, p. 593–602.
5. *D.A. Donciulescu, F.G. Filip* (1983). **DISPATCHER : a decision support system based on hierarchical approach.** 5 th Intern. Conf. on Contr. Syst. and Comp. Sci, Bucureşti vol. I, p. 189–194.

5. APLICAŢII NESTANDARD ALE SISTEMELOR MARI

În cadrul cercetărilor în domeniul conducerii sistemelor mari, o preocupare importantă se referă la stabilirea modelului matematic folosit. În majoritatea cazurilor, pentru sistemele mari nu se dispune de model matematic, sau modelul matematic disponibil este de dimensiuni mari, ceea ce ridică probleme în alegerea metodei de conducere şi a configuraţiei de echipamente folosite (în sensul unui cost ridicat al instrumentaţiei de măsurare şi control). O serie de lucrări ale *secţiunii 11.1/E* tratează aspecte legate de această problemă.

Astfel, Davison şi Solomon (11.1/E4) tratează problema alegerii structurii de comandă pentru reglarea robustă a sistemelor multivariabile mari al căror model matematic nu este cunoscut. Selectarea intrărilor care urmează a fi folosite pentru comanda sistemului şi a tipului structurii de comandă este una dintre cele mai critice probleme ale conducerii sistemelor

mari. Lucrarea tratează această problemă pentru procese care pot fi descrise de modele lineare invariante, de tipul ecuaţiei de stare şi ieşire următoare:

$$\begin{aligned} \dot{x} &= Ax + Bu + E\omega \\ y &= Cx + Du + F\omega, \quad e \overset{\Delta}{=} y - y_{ref} \end{aligned} \tag{1}$$

unde $x \in R^n$ este vectorul de stare, $u \in R^m$ este vectorul intrărilor, $y \in R^r$ este vectorul ieşirilor, $e \in R^r$ este vectorul erorii în sistem, y_{ref} este vectorul referinţelor, iar $\omega \in R^{\Omega}$ este vectorul perturbaţiilor în sistem.

Se consideră cazul proceselor stabile în buclă deschisă. În aceste condiţii, sînt prezentate o serie de structuri de comandă (distincte pentru cazurile perturbaţiilor şi referinţelor constante, sau de tip rampă), cuprinzînd matrici care pot fi totdeauna determinate pe baza unor experimente realizate în regim staţionar. Rezultă că nu este necesară cunoaşterea modelului matematic al procesului nici pentru verificarea condiţiilor de existenţă a soluţiei problemei de reglare robustă a procesului respectiv, nici pentru implementarea structurilor de comandă.

Structurile de comandă sînt parametrizate prin intermediul unor scalari, care sînt determinaţi prin acordarea on-line a regulatoarelor. Configuraţiile de comandă obţinute prin aplicarea structurilor de comandă prezentate în lucrare pot fi complet centralizate, parţial descentralizate sau complet descentralizate, în funcţie de structura matricilor obţinute experimental.

Se arată că dacă matricea funcţie de transfer evaluată la frecvenţă zero, este dominant diagonală, atunci, pentru rezolvarea problemei servomecanismului robust poate fi folosită totdeauna o structură de comandă descentralizată. De asemenea sînt prezentate o serie de exemple (comanda unui furnal cu 4 intrări şi 4 ieşiri, a unei coloane de distilare cu 3 intrări şi 3 ieşiri şi a unei instalaţii chimice cu 5 intrări şi 4 ieşiri), pentru ilustrarea rezultatelor obţinute prin aplicarea structurilor de comandă prezentate în lucrare, unor procese al căror model matematic nu este cunoscut.

Hodzic şi Šiljak (11.1/E2) prezintă o metodă de proiectare linear pătratică gaussiană (LPG) bazată pe teoria sistemelor ierarhizate, care permite reducerea considerabilă atît a calculelor off-line, cît şi a celor on-line.

O metodă standard de reducere a dimensiunii problemei, în cazul sistemelor algebrice rare mari, este reordonarea variabilelor şi ecuaţiilor pentru a aduce sistemul în forma bloc-triunghiulară. Sistemul global poate fi apoi rezolvat ca o secvenţă de subsisteme de dimensiuni mai mici, rezultînd o scădere considerabilă a cerinţelor de calcul. Folosind această metodă, a fost dezvoltat un algoritm bazat pe teoria grafurilor, care poate fi folosit pentru transformarea unui sistem dinamic într-un sistem ierarhizat format din subsisteme structural observabile şi controlabile. Pentru această structură, lucrarea prezintă o metodă de proiectare a unor estimatoare şi regulatoare de ordin mic, pentru fiecare subsistem separat, pornind de sus în jos în cadrul structurii ierarhizate.

Se consideră sistemul discret $\mathcal{S}$ compus din s subsisteme:

$$\begin{aligned} \mathcal{S}: x_i(t+1) &= \sum_{j=1}^{i} A_{ij}x_j(t) + \sum_{j=1}^{i} \Gamma_{ij}w_j(t) \\ y_i(t) &= \sum_{j=1}^{i} C_{ij}x_j(t) + v_i(t) \\ i &= 1, 2, \ldots, s \end{aligned} \tag{2}$$

care este reprezentată de o structură ierarhizată (bloc triunghiulară), compusă din subsistemele:

$$\begin{aligned} \mathcal{S}_i: x_i(t+1) &= A_{ii}x_i(t) + \Gamma_{ii}w_i(t) \\ y_{ii}(t) &= C_{ii}x_i(t) + v_i(t) \\ i &= 1, 2, \ldots, s \end{aligned} \tag{3}$$

unde $x_i(t) \in R^{n_i}$, $y_i(t) \in R^{l_i}$ sînt vectorii de stare respectiv de ieşire ai subsistemului $\mathcal{S}_i$ la momentul $t=0, 1, 2, \ldots$; $w_i(t) \in R^{r_i}$ şi $v_i(t) \in R^{l_i}$ sînt procese gaussiene-independente cu media zero şi covarianţele R_w^{ii} şi respectiv R_v^{ii}; A_{ij}, Γ_{ij}, C_{ij} sînt matrici constante de dimensiuni corespunzătoare. Sistemul (2) poate fi descris şi într-o formă compactă:

$$\mathcal{S}: \begin{aligned} x(t+1) &= Ax(t) + \Gamma \cdot w(t) \\ y(t) &= Cx(t) + v(t) \end{aligned} \tag{4}$$

unde $x(t) \in R^n$, $y(t) \in R^l$ sînt vectorii de stare respectiv de ieşire, ai sistemului $\mathcal{S}$; $w(t) \in R^r$, $v(t) \in R^l$ reprezintă perturbaţiile, respectiv zgomotul de măsurare corespunzătoare lui $\mathcal{S}$. Matricile sistemului A, Γ şi C sînt bloc triunghiulare. Se urmăreşte proiectarea unui estimator ierarhizat pentru sistemul $\mathcal{S}$, sub forma:

$$\hat{\mathcal{S}}^{\oplus}: \begin{aligned} \hat{x}(t+1 \mid t) &= A\hat{x}(t \mid t-1) + K^{\oplus}\bar{y}(t) \\ \bar{y}(t) &= y(t) - C\hat{x}(t \mid t-1) \end{aligned} \tag{5}$$

matricea de amplificare $K^{\oplus}$ avînd forma bloc triunghiulară:

$$K^{\oplus} = \begin{bmatrix} K_{11}^{\oplus} & & & \bigcirc \\ K_{21}^{\oplus} & K_{22}^{\oplus} & & \\ \vdots & & \ddots & \\ K_{s1}^{\oplus} & K_{s2}^{\oplus} & \ldots & K_{ss}^{\oplus} \end{bmatrix} \tag{6}$$

Definind estimările actualizate sub forma:

$$\hat{x}_i(t \mid t) = \hat{x}_i(t \mid t-1) + K_{ii}^{*}[y_i(t) - \sum_{j=1}^{i} C_{ij}\hat{x}_j(t \mid t) - C_{ii}\hat{x}_i(t \mid t-1)] \quad i=1, 2, \ldots, s \tag{7}$$

rezultă:

$$\hat{x}_i(t+1 \mid t) = \sum_{j=1}^{i} A_{ij}\hat{x}_j(t \mid t), \quad i=1, 2, \ldots, s \tag{8}$$

Prin transcrierea ecuaţiei (7) prin introducerea termenilor $\bar{y}_j(t)$ şi formînd matricea bloc triunghiulară K^*, se obţine (8) în forma compactă:

$$\hat{x}(t \mid t) = \hat{x}(t \mid t-1) + K^*\bar{y}(t) \tag{9}$$

Această formă corespunde estimatorului optimal global, dar cum K^* va fi calculat folosind o metodă de optimizare ierarhizată, estimatorul nu este optimal în general. Eroarea de estimare $\tilde{x}(t \mid t) = x(t) - \hat{x}(t \mid t)$ este dată de:

$$\tilde{x}(t \mid t) = (I - K^*C)\tilde{x}(t \mid t-1) - K^*v(t) \tag{10}$$

Folosind ecuaţiile estimatorului exprimate în funcţie de $K^{\oplus}$ şi K^*, se obţine:

$$K_{ii}^{\oplus} = A_{ii}K_{ii}^{*}, \quad K_{ij}^{\oplus} = \sum_{k=j}^{i} A_{ik}K_{kj}^{*}, \quad i > j \tag{11}$$

$$K^{\oplus} = AK^* \tag{12}$$

Eroarea de estimare $\tilde{x}(t+1 \mid t)=x(t)-\tilde{x}(t+1 \mid t)$ poate fi scrisă sub forma :

$$\tilde{x}(t+1 \mid t)=A\tilde{x}(t \mid t)+\Gamma w(t) \tag{13}$$

Din ecuaţia (10) rezultă că matricea de covarianţă a erorii de actualizare staţionare este de forma :

$$P^{+}=(I-K^{*}C)P^{-}(I-K^{*}C)^{T}+K^{*}R_{v}K^{*T} \tag{14}$$

unde P^{-} reprezintă matricea de covarianţă a erorii de predicţie staţionare :

$$P^{-}=AP^{+}A^{T}+\Gamma R_{w}\Gamma^{T} \tag{15}$$

Fiecărui subsistem S_i i se asociază un indice de performanţă de forma :

$$J_i=\mathrm{tr}P_{ii}, \quad i=1,2,\ldots,s \tag{16}$$

Aceşti indici de performanţă sînt optimizaţi separat, pentru fiecare subsistem, pornind de sus în jos în structura ierarhizată a sistemului.

Această procedură nu conduce, desigur, la un estimator global optimal, dar acest estimator este suboptimal şi stabil, în condiţii destul de lejere.

Se obţine :

$$K^{*}_{ii}=\left(\sum_{j=1}^{i} P_{ij}C_{ij}\right)\left[\sum_{j=1}^{i} C^{*}_{ij}\sum_{j=1}^{i} P^{-}_{jk}C^{*T}_{ik}+\left(\sum_{j=1}^{i} D^{*}_{ij}\right)R_{v}\left(\sum_{j=1}^{i} D_{ij}\right)^{T}\right]^{-1} \tag{17}$$

unde :

$$C_{ij}=\begin{cases} C_{ii}, & i=j \\ -\sum_{k=j}^{i-1}\sum_{l=k}^{i-1} C_{il}K_{lk}C_{kj}+C_{ij}, & i>j \end{cases} \tag{18}$$

$$D^{*}_{ij}=\begin{cases} D_{ii}, & i=j \\ -\sum_{k=j}^{i-1} C_{ik}K_{kj}D_{jj}, & i>j \end{cases}$$

Metoda de proiectare ierarhizată a estimatoarelor prezentată în lucrare este o generalizare a unei metode propuse de Pichai, Sezer şi Šiljak.

Datorită structurii bloc triunghiulare a estimatorului $\hat{\mathcal{S}}^{\oplus}$, stabilitatea fiecărui estimator de subsistem, $\hat{\mathcal{S}}_i^{\oplus}$, implică stabilitatea estimatorului global, descris de ecuaţia (5), care poate fi rescrisă sub forma :

$$\hat{\mathcal{S}}^{\oplus}: \hat{x}(t+1 \mid t)=(A-K^{\oplus}C)\hat{x}(t \mid t-1)+K^{\oplus}y(t) \tag{19}$$

Algoritmul bazat pe teoria grafurilor, folosit pentru a reprezenta sistemulor original $\mathcal{S}$ sub forma ierarhizată (3), poate demonstra că fiecare pereche (A_{ii}, C_{ii}) este structural observabilă şi $[A_{ii}, \Gamma_{ii}(R_w^{ii})^{1/2}]$ este structural controlabilă. Acest fapt, ca şi metoda de optimizare ierarhizată, garantează că toate estimatoarele subsistemelor $\hat{\mathcal{S}}_i^{\oplus}$ sînt stabile, ceea ce asigură stabilitatea lui $\hat{\mathcal{S}}^{\oplus}$.

Pentru problema de comandă stohastică, se consideră sistemul $\mathcal{S}$ descris de :

$$\begin{aligned} \mathcal{S}: x_i(t+1)&=\sum_{j=1}^{i} A_{ij}x_j(t)+\sum_{j=1}^{i} B_{ij}u_j(t)+w_j(t) \\ y_i(t)&=\sum_{j=1}^{i} C_{ij}x_j(t)+v_i(t) \\ i&=1,2,\ldots,s \end{aligned} \tag{20}$$

unde $u(t) \in R^{m_i}$ sînt intrările deterministe, sau :

$$\mathcal{S}: x(t+1) = Ax(t) + Bu(t) + w(t)$$
$$y(t) = Cx(t) + v(t) \tag{21}$$

unde B are aceeaşi structură bloc triunghiulară ca şi Γ. Indicele de performanţă asociat sistemului $\mathcal{S}$ este descris de :

$$J_i = \lim_{T\to\infty} \frac{1}{T} \left\{ \sum_{t=0}^{T-1} x_i^T(t)\, Q_x^{ii} x_i(t) + u_i^T(t)\, Q_u^{ii} u_i(t)] \right\} \tag{22}$$

$$i = 1, 2, \ldots, s$$

unde matricile constante Q_x^{ii} sînt ne-negativ definite, iar Q_u^{ii} sînt pozitiv definite.

Metoda de descompunere folosită, bazată pe teoria grafurilor, asigură satisfacerea automată a condiţiilor de controlabilitate structurală şi observabilitate structurală a perechilor (A_{ii}, B_{ii}), respectiv $[A_{ii}, (Q_x)^{1/2}]$. Urmează un proces de optimizare secvenţială a subsistemelor $\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_s$, obţinîndu-se vectorul de comandă global $u^*(t) = [u_1^{*T}(t), u_2^{*T}(t), \ldots, u_s^{*T}]^T$ sub forma :

$$u^*(t) = -L^* \hat{x}(t \mid t) \tag{23}$$

unde $\hat{x}(t \mid t)$ este definit în (9), iar matricea de amplificare L^* este bloc triunghiulară. În lucrare este demonstrată stabilitatea sistemului global obţinut prin metoda de proiectare LPG ierarhizată prezentată.

În (11.1/E6), Jamshidi şi Wang abordează problema conducerii sistemelor mari cu întîrzieri, pe baza teoriei sistemelor ierarhizate, cu aplicaţii în domeniul utilizării resurselor de apă. Această problemă a mai fost abordată pe baza principiului maximului. Sistemul mare nelinear cu multiple întîrzieri este descris sub forma :

$$\dot{X} = F(x(t), x(t-\tau_1), \ldots, x(t-\tau_p), U(t-h_1), \ldots, U(t-h_q)) \tag{24}$$

$$t \geqslant t_0, \qquad \tau_p \geqslant \ldots \geqslant \tau_i \geqslant \ldots \geqslant \tau_1$$
$$h_q \geqslant \ldots \geqslant h_i \geqslant \ldots \geqslant h_1$$
$$X(t) = X_0(t) \qquad t_0 - \tau_p \leqslant t \leqslant t_0$$
$$U(t) = U_0(t) \qquad t_0 - h_q \leqslant t \leqslant t_0$$

unde $X(t) \in R^n$ şi $U(t) \in R^m$ sînt vectorii de stare, respectiv de comandă, τ_i, $i = 1, 2, \ldots, p$ şi h_j, $j = 1, 2, \ldots, q$ sînt constante pozitive reprezentînd întîrzierile. F (.) este o funcţie continuu diferenţiabilă în raport cu argumentele, t_0 este momentul iniţial iar $X_0(t)$ şi $U_0(t)$ sînt funcţii iniţiale specificate. Indicele de performanţă de minimizat este :

$$J = 1/2 X^T(t_f)\, S X(t_f) + 1/2 \int_{t_0}^{t_f} [X^T(t)\, Q(t)\, X(t) + U^T(t)\, R(t)\, U(t)]\, dt \tag{25}$$

unde S şi Q sînt matrici diagonale pozitiv semidefinite, iar matrice R (t) este diagonală pozitiv definită şi continuă pe porţiuni. Se pune problema determinării vectorului de comandă $U(t)$, $t_0 \leqslant t \leqslant t_f$, care la momentul final t_f, avînd starea finală liberă $X(t_f)$, să satisfacă (24) şi să minimizeze indicele de performanţă (25).

Modelul sistemului (24) este linearizat folosind metoda dezvoltării în serii Taylor în jurul punctului nominal (X_n, U_n).

Se obţine următoarea aproximaţie lineară a sistemului :

$$\dot{x}(t) = A_0(t)\, x(t) + \sum_{k=1}^{p} A_k(t)\, x(t-\tau_k) + B_0(t)\, u(t) + \sum_{l=1}^{q} B_l(t)\, u(t-h_l) \tag{26}$$

cu traiectoriile inițiale:

$$x(t)=x_0(t)-x_n(t)=x_0(t) \ldots t_0-\tau_p\leq t\leq t_0$$
$$u(t)=U_0(t)-U_n(t)=u_0(t) \ldots t_0-\tau_q\leq t\leq t_0$$

iar indicele de performanță (27) devine:

$$\hat{J}=\frac{1}{2}x^T(t_f)(Sx(t_f)+2SX_n(t_f))+\frac{1}{2}\int_{t_0}^{t_f}[x^T(Q(t)x(t)+2Q(t)X_n(t))+$$
$$+u^T(t)(R(t)u(t)+2R(t)U_n(t))]\,dt \tag{27}$$

Sistemul este descompus în N subsisteme, prin partiționarea vectorului de stare x și al celui de comandă, u, (de cele mai multe ori, în cazul sistemelor mari, descompunerea în subsisteme a sistemului global, e naturală):

$$\dot{x}_i(t)=A_{0i}(t)x_i(t)+B_{0i}(t)u_i(t)+Z_i(t) \quad t\geq t_0 \tag{28}$$
$$x_i(t)=x_{i0}(t) \quad \tau_p\leq t\leq t_0$$
$$u_i(t)=u_{i0}(t) \quad h_q\leq t\leq t_0$$

$$z_i(t)=\hat{A}_{0i}(t)x(t)+\hat{B}_{0i}(t)u(t)+\sum_{k=1}^{p}A_{ki}(t)x(t-\tau_k)+\sum_{l=1}^{q}B_{li}(t)u(-h_l)=z_{ie}(t) \tag{29}$$

unde: $\hat{A}_{0i}$=Bloc—diag $[A_{i1}, A_{i2}, \ldots, A_{i,i-1}, 0, A_{i,i+1}, \ldots, A_{iN}]$. Se consideră de asemenea că:

$$\min_u \hat{J}=\sum_{i=1}^{N}\min \hat{J}_i \tag{30}$$

unde:

$$\hat{J}_i=\frac{1}{2}x_i^T(t_f)(S_ix_i(t_f))+\frac{1}{2}\int_{t_0}^{t_f}[x_i(t)+(Q_i(t)x_i(t)+2Q_i(t)x_{ni}(t))+$$
$$+u_i^T(t)(R_i(t)u_i(t)+2R_i(t)U_{ni}(t))]\,dt \tag{31}$$

Lucrarea propune un algoritm de determinare a comenzii suboptimale a problemei definite de (26) și (27), pe baza metodei predicției interacțiunilor. Termenii de interacțiune și cei de întîrziere sînt considerați ca intrări de perturbație, obținîndu-se subsisteme fără întîrzieri. În cadrul structurii de calcul ierarhizate obținute, la nivelul coordonatorului se realizează simple înlocuiri, ceea ce facilitează folosirea metodei. În lucrare este prezentat un exemplu de aplicare a algoritmului în cazul unui sistem de utilizare a resurselor de apă.

În continuare, sînt prezentate pe scurt celelalte lucrări ale acestei secțiuni.

Mohler, Kolodziej și Bugnen analizează în [11.1E1] modele pentru sisteme bilineare cuplate, care, în unele cazuri, apar în mod natural, sau sînt folosite ca aproximări convenabile ale unor sisteme de nelinearități mai mari.

Santiago, Lange și Jamshidi prezintă în [11.1E3] preocupări legate de modelarea și comanda structurilor spațiale flexibile mari. Lucrarea prezintă problemele de bază, caracteristicile și dificultățile inerente în modelarea și comanda unor astfel de structuri.

În [11.1E5] este prezentat un sistem întrebare-răspuns pentru interfețe pentru baze de date inteligente. Relațiile de similaritate a datelor dintr-o bază de date avînd un model bazat pe similaritate psihologică, sînt reprezentate sub formă de rețea. O funcție de generalizare permite propagarea în rețea a informației obținute din răspunsurile de similaritate ale utilizatorului. Folosind funcția de generalizare, calculatorul determină tipul generic de întrebare pe care trebuie să o pună în continuare utilizatorului, în scopul folosirii eficiente a informației. Utilizatorului i se permite să dea răspunsuri nedefinite sau imprecise, care reflectă lipsa de cunoștințe. Prin acest proces de întrebare-răspuns, sistemul bazat pe cunoștințe facilitează specificarea de către utilizator a unor cereri de date relevante, ca și regăsirea de date din baza de date.

ANALIZA ŞI PROIECTAREA ASISTATE DE CALCULATOR A SISTEMELOR

Dr. ing. *A. Varga*
Ing. *M. Dumitru*
Ing. *A. Moangă*
I.T.C.I.

PACHETE DE PROGRAME DE PROIECTARE ASISTATĂ DE CALCULATOR

În secţiunea a 11.2/A se prezintă cîteva pachete de programe destinate problematicilor asociate proiectării asistate de calculator a sistemelor automate. Lucrările [11.2/A4] şi [11.2/A5] prezintă pachete interactive pentru identificarea experimentală a proceselor tehnologice. Pachetul descris în lucrarea [11.2/A3] este destinat determinării modelelor de ordin redus, ca şi pentru validarea modelelor rezultate. Un pachet interactiv pentru proiectarea sistemelor în domeniul frecvenţă utilizînd metoda sintezei totale este descris în [11.2/A2].

Lucrarea [11.2/A1] descrie un sistem integrat de pachete de programe interactive care acoperă aspectele de identificare, analiză, proiectare şi simulare şi oferă facilităţi pentru implementarea on-line a rezultatelor proiectării. Lucrarea [11.2/A6] prezintă o bibliotecă de subprograme FORTRAN portabile, care implementează o serie de algoritmi numerici performanţi pentru rezolvarea unor probleme matematice de bază ce intervin în analiza, modelarea şi proiectarea sistemelor automate multivariabile. Vom prezenta în cele ce urmează pe scurt caracteristicile principale ale acestor pachete.

Realizarea unei aplicaţii de conducere automată cu calculator a unui proces tehnologic necesită în general parcurgerea a patru etape succesive: (1) modelarea procesului; (2) proiectarea sistemului de conducere automată; (3) implementarea algoritmilor de conducere numerică directă pe calculatorul de proces; şi (4) analiza performanţelor sistemului de conducere proiectat. Pentru realizarea primelor două etape se pot utiliza pachete de programe interactive de modelare şi proiectare asistată de calculator. Parcurgerea etapei a treia presupune elaborarea programelor (de regulă în limbajul de asamblare a calculatorului de proces) care efectuează calculele specifice algoritmilor de conducere. Etapa a patra realizează atît o testare a corectitudinii programelor de conducere, cît şi o validare a rezultatelor proiectării în condiţiile de funcţionare în timp real a sistemului de conducere cu calculatorul. De regulă etapele (2)÷(4) se parcurg repetat, pînă la găsirea valorilor optime ale parametrilor care apar în algoritmii de conducere.

Lucrarea 11.2/A1 prezintă un sistem integrat de pachete de programe interactive, denumit CACOS, care realizează pe lîngă funcţiunile specifice etapelor de modelare şi proiectare asistate de calculator şi pe aceea de generare automată a programelor de conducere în limbajul de asamblare al minicalculatorului NOVA3. Sistemul CACOS are la bază pachetele MAT (Calcule matriciale), TS (Serii de timp), PIPACK (Identificarea parametrilor) şi DPACS (Analiza şi proiectarea sistemelor) cu ajutorul cărora se pot efectua funcţiile de modelare şi proiectare. Generarea automată a codului sursă pentru programul de conducere se face cu programul PCODE. Pentru obţinerea unui timp de calcul cît mai redus, PCODE generează instrucţiunile de adunare şi înmulţire în virgulă mobilă numai pentru elemente nenule ale matricilor care intervin în algoritmii de conducere. Codul generat de PCODE este compilat şi înglobat în programul RCOR care efectuează funcţia de conducere în timp real. Alte funcţiuni pe care

RCOR le poate realiza sînt : culegere de date, afişare parametri, listare serii de timp, introducere valori pentru referinţe etc.

Sistemul CACOS a fost utilizat în proiectarea unui sistem automat de levitaţie magnetică. Timpul necesar proiectării şi experimentării (aproximativ 1 oră) a fost mult mai mic decît timpul necesar codificării manuale a algoritmilor de conducere. Soluţia adoptată în sistemul CACOS reprezintă una dintre căile de promovare a rezultatelor noi ale teoriei sistemelor în aplicaţii practice.

În *lucrarea 11.2/A2* este prezentat un pachet interactiv, denumit TSPSP, care implementează o metodologie de proiectare în domeniul frecvenţă. Problema de proiectare, denumită *problema sintezei totale* (PST), se formulează în termenii a trei aplicaţii lineare fundamentale: P, M şi T definite după cum urmează

$$P : U(s) \rightarrow Y(s)$$

$$M : R(s) \rightarrow U(s)$$

$$T : R(s) \rightarrow Y(s)$$

în care U (s), Y (s) şi R (s) sînt spaţii vectoriale raţionale ale transformatelor (Laplace sau Z) ale intrărilor, ieşirilor şi referinţelor, respectiv. Pentru baze fixate, P, M şi T se reprezintă prin matrici de transfer avînd următoarele semnificaţii : P (s) este matricea de transfer a instalaţiei, M (s) este matricea de transfer dintre intrări şi referinţe, iar T (s) este matricea de transfer dintre ieşiri şi referinţe. Între aceste matrici există relaţia evidentă

$$T(s) = P(s)M(s)$$

Ecuaţia de mai sus se poate reformula ca o problemă de calcul al nucleului unei aplicaţii, în forma

$$[P \quad -I]\begin{bmatrix} M \\ T \end{bmatrix} = 0$$

PST constă în alegerea perechilor (M, T) din nucleul aplicaţiei [P —I], astfel încît să satisfacă anumite cerinţe de proiectare şi realizabilitate fizică. Stabilitatea internă impune ca matricile M (s) şi T (s) să fie stabile şi proprii (gradele numitorilor mai mari decît sau egale cu gradele numărătorilor). Prin manipularea bazei calculate pentru nucleu, se determină (M, T) astfel încît să conducă la răspunsuri ale ieşirilor şi la comenzi asociate, dorite de proiectant.

După ce M (s) şi T (s) au fost determinate, proiectarea continuă prin găsirea structurii de comandă adecvate. Configuraţia generală este prezentată în fig. 1, unde G şi H sînt matrici de transfer care se aleg astfel încît să-l realizeze pe M.

Din relaţia intrare-ieşire rezultată din fig. 1, obţinem ecuaţia de bază care trebuie satisfăcută

$$(I+PGH)^{-1}PG = PM$$

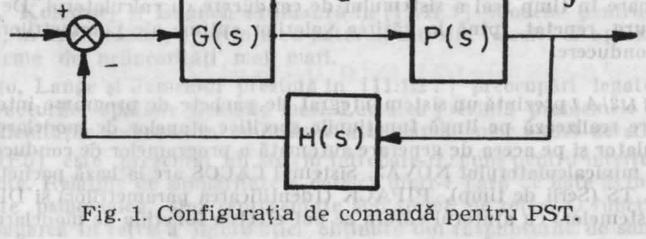


Fig. 1. Configuraţia de comandă pentru PST.

Obiectivul proiectantului este de a distribui dinamica din M, în G şi H astfel încît să rezulte o sinteză care este insensibilă la variaţii parametrice în instalaţie. Dacă alegem H = O şi G = M, structura de comandă este în buclă deschisă. O altă posibilitate este alegerea lui H = I, deci utilizarea unei reacţii unitare. Se obţine imediat

$$G = M(I - PM)^{-1}$$

O altă posibilitate de alegere, bazată pe minimizarea sensibilităţii sistemului în buclă închisă, este descrisă în lucrare.

Pachetul TSPSP constă dintr-un număr de programe care realizează calculele implicate de PST. Astfel, sînt prevăzute programe de calcul a bazei nucleului unui operator raţional sau polinomial, de manipulare a bazei, de calcul cu matrici de transfer, de realizare minimală. Verificarea proiectării se face prin simulare. Comunicarea între programe se face prin fişiere care utilizează un format comun de memorare a datelor. Programele sînt interactive şi permit efectuarea proiectării plecînd de la descrierea originală a instalaţiei şi terminînd cu obţinerea structurii finale a compensatorului.

În *lucrarea 11.2/A3* este descris un pachet interactiv de programe pentru determinarea modelelor de ordin redus ale sistemelor complexe. Pachetul de programe are la bază o bibliotecă de subprograme FORTRAN ce implementează principalele metode cunoscute de reducere a ordinului în domeniul frecvenţă sau în domeniul timp. O altă bibliotecă de subprograme este destinată testării calităţii modelului redus, în comparaţie cu modelul original. O rutină specială este prevăzută pentru predicţia ordinului modelului pornind de la criterii de reducere deterministe sau utilizînd proceduri iterative.

Funcţiile principale ale pachetului de programe sînt :

1) transformări între reprezentările de stare şi reprezentările în frecvenţă ;

2) determinarea ordinului modelului redus ;

3) determinarea modelului de ordin redus printr-una dintre metodele de reducere disponibile ;

4) simularea sistemului redus pentru semnale de intrare date şi analiza în domeniul frecvenţă a modelului redus ;

5) evaluarea valabilităţii procedurii de reducere utilizate.

În lucrare se prezintă o aplicaţie de reducere a ordinului unui model al unui sistem energetic folosind atît metode în domeniul timp, cît şi în frecvenţă. Modelul redus a fost determinat în vederea proiectării unui regulator suboptimal de ordin cît mai mic.

Pachetul este implementat pe un calculator WAX 11/780.

Lucrarea 11.2/A4 prezintă pachetul de programe interactive SATER, destinat identificării sistemelor cu o intrare şi o ieşire. Acest pachet a fost dezvoltat în primul rînd în scop didactic, pentru a se pune la dispoziţia studenţilor o gamă largă de metode de identificare în vederea comparării acestora. Datele necesare pentru determinarea modelelor se pot obţine fie prin simulări, fie prin culegere directă din procesul real. Pachetul are o structură modulară şi se poate extinde relativ uşor cu alte programe.

Pachetul SATER are trei componente principale : supervizorul, programele aplicative şi rutinele de serviciu. *Supervizorul* constă dintr-un executiv care are rolul de a lansa programul aplicativ selectat de utilizator şi dintr-un modul de interogare-interpretare, care solicită datele şi parametrii necesari rulării programului aplicativ dorit. *Programele aplicative* formează partea cea mai mare a pachetului SATER şi implementează diferite metode de estimare a parametrilor, de determinare a ordinului modelului, de analiză, de simulare etc. *Rutinele de serviciu* efectuează operaţiile de intrare-ieşire şi includ subsistemul pentru întrebări-răspunsuri şi subsistemul pentru grafică pe terminal sau ploter.

Vom prezenta în continuare principalele operaţii implementate în pachetul SATER.

a) Culegerea şi prelucrarea primară a datelor, analiza semnalelor

— eşantionare pe maximum 8 canale ale sistemului de culegere DEC LPS

— filtrarea datelor cu filtru Butterworth de ordinul 3

— prelucrarea statistică a semnalelor

— analiza armonică a semnalelor

— analiza spectrală

b) Generarea datelor simulate

Se pot utiliza şase tipuri de semnale pentru simulări

— semnale sinusoidale

— semnale polinomiale (impuls, treaptă, rampă etc.)

— zgomot alb uniform

— secvenţă maximală

— zgomot alb gaussian

— serii de timp arbitrare memorate

c) Teste pentru ordinul modelului, validare

Testele bazate pe rezultatele estimării parametrilor includ :

— simplificare poli-zerouri

— netezimea ajustărilor recursive ale parametrilor pentru ordine crescătoare ale modelului

— reziduali, erori de predicţie, funcţii de eroare

— testul abaterii semnalelor

— testul produsului matricilor de moment.

Testele care nu utilizează estimarea parametrilor includ :

— funcţia de pierdere (Woodside)

— testul de singularitate a produsului matricilor de momente ale datelor.

d) Metode de estimare a parametrilor, estimarea marginii Cramér-Rao

— metoda matricii extinse

— metoda celor mai mici pătrate generalizate

— metoda variabilelor istrumentale

— metoda verosimilităţii maxime

e) Conversii ale modelelor discrete în modele continue

— cu extrapolator de ordin zero

— cu extrapolator de ordin unu

— cu extrapolator triunghiular

— transformare — Z bilineară

f) Analiza proceselor discrete şi continue

— calculul poli-zerourilor

— trasarea diagramelor Nyquist sau Bode pentru sisteme continue

— trasarea locului rădăcinilor

Pachetul SATER este implementat pe un minicalculator DEC PDP 11/60 sub sistemul de operare RSX 11M.

În *lucrarea 11.2/A5* se prezintă pachetul IP-DEM destinat identificării sistemelor cu mai multe intrări şi o ieşire, utilizîndu-se descrierea prin ecuaţii cu diferenţe. Pachetul este format din trei programe independente. P-1 efectuează culegerea şi prelucrarea primară (statistică) a datelor. Semnalele de perturbaţie ale procesului pot fi generate manual sau automat, sub formă de semnale pseudoaleatoare binare. P-2 efectuează identificarea modelului utilizînd datele rezultate de la P-1Sint disponibile patru metode de identificare : (1) metoda variabilelor instrumentale cu ieşiri întîrziate ; (2) metoda celor mai mici pătrate ; (3) metoda filtrării neliniare aproximative ; şi (4) metoda verosimilităţii maxime. Modelele obţinute cu P-2 se pot compara cu programul P-3 în vederea selectării modelului celui mai adecvat.

Pachetul IP-DEM este implementat pe un microcalculator DEC LSI 11/23 şi este scris în limbajul FORTRAN. În lucrare este demonstrată utilizarea pachetului atît în cazul unui model simulat cît şi în cazul determinării modelului unei coloane de distilare binare.

Lucrarea 11.2./A6 prezintă pachetul de programe BIMAS elaborat în cadrul Institutului pentru tehnică de calcul şi informatică din Bucureşti. BIMAS este un pachet de subprograme FORTRAN, destinat rezolvării numerice a problemelor matematice de bază care intervin în analiza şi proiectarea asistată de calculator a sistemelor automate lineare. Algoritmii implementaţi au fost riguros selectaţi în vederea satisfacerii atributelor de generalitate, fiabilitate, stabilitate numerică, precizie şi eficienţă. BIMAS are la bază pachetele bine cunoscute de algebră liniară BLAS, LINPACK şi EISPACK, şi poate fi considerat ca o extindere a acestor pachete la probleme de calcul matricial. BIMAS constă din aproximativ 50 de subprograme originale sau adaptate din literatura de specialitate şi circa 25 de subprograme din pachetele amintite mai sus.

Pachetul BIMAS conţine subprograme pentru următoarele probleme :

— calculul şi ordonarea formei Schur reale (FSR) a unei matrici pătrate ;

— calculul şi ordonarea formei Schur reale generalizate a unei perechi de matrici pătrate ;

— calculul formei bloc-diagonale a unei matrici ;

— rezolvarea ecuaţiilor matriciale lineare de tip Sylvester şi Liapunov ;

— stabilizarea sistemelor lineare multivariabile ;

— alocarea polilor sistemelor lineare multivariabile ;

— rezolvarea ecuaţiilor matriciale algebrice de tip Riccati ;

— calculul exponenţialelor matriciale.

Vom prezenta pe scurt cîteva caracteristici ale pachetului BIMAS. Majoritatea algoritmilor implementaţi utilizează FSR a unei matrici, calculată prin transformări ortogonale. Utilizarea largă a acestor transformări măreşte fiabilitatea calculelor numerice. Pentru multe probleme, programele evaluează numere de condiţionare adecvate sau informaţii echivalente.

În elaborarea pachetului s-a utilizat o abordare modulară pentru rezolvarea problemelor complexe. Majoritatea calculelor implică efectuarea unei secvenţe de apeluri la subrutinele pachetului. Modularitatea sporeşte capabilităţile de segmentare, astfel încît pachetul se poate utiliza eficient chiar pe minicalculatoare.

Portabilitatea pachetului a fost asigurată prin utilizarea limbajului FORTRAN standard, fără să se exploateze facilităţile oferite de implementări particulare ale compilatoarelor. În acest scop, toţi parametrii dependenţi de maşină, cum sînt de exemplu precizia relativă sau radix-ul maşinii, sînt furnizaţi prin intermediul a două rutine speciale. La transportarea pachetului de pe un echipament pe altul, doar aceste rutine sînt modificate.

Pachetul BIMAS a fost dezvoltat pe minicalculatorul CORAL-4011 în dublă precizie. Subprogramele sînt documentate extensiv, într-un mod unitar. Pentru toate rutinele de bază s-au elaborat programe de test. Pachetul BIMAS a fost dezvoltat în continuare prin realizarea pachetului de subprograme BIMASC, destinat rezolvării problemelor specifice de calcul ce intervin în modelarea, analiza, proiectarea şi simularea sistemelor lineare multivariabile. Utilizîndu-se subprogramele pachetelor BIMAS şi BIMASC, s-a elaborat (tot în cadrul ITCI) un puternic pachet interactiv, CASAD, destinat proiectării asistate de calculator a sistemelor automate. CASAD este implementat pe minicalculatoarele româneşti şi este unul dintre pachetele cele mai performante de proiectare în domeniul timp existente pe plan mondial.

Concluzii. Din prezentarea pachetelor, se evidenţiază unele tendinţe existente pe plan mondial în elaborarea de software pentru proiectarea asistată de calculator a sistemelor automate:

1) elaborarea de pachete interactive care implementează metodologii complete de identificare, analiză sau proiectare, utilizînd tehnici numerice noi;

2) utilizarea unor algoritmi robuşti şi a unui software numeric de calitate în rezolvarea problemelor specifice de automatică;

3) constituirea unor biblioteci de subprograme FORTRAN portabile pentru rezolvarea problemelor de automatică, elaborate pe baza unor standarde de programare, utilizare, testare şi documentare;

4) implementarea pachetelor pe echipamente de calcul de tip mini- şi micro-calculatoare, cu folosirea intensivă a unor facilităţi grafice.

Menţionăm că ţara noastră se află printre ţările avansate în domeniul elaborării de pachete de programe destinate proiectării asistate de calculator a sistemelor automate.

METODE DE PROIECTARE ASISTATĂ DE CALCULATOR A SISTEMELOR AUTOMATE

În *secţiunea 11.2/C* se prezintă cîteva metode utilizate în analiza şi proiectarea asistată de calculator a sistemelor automate. Lucrarea [11.2/C1] prezintă unele tehnici de bază pentru analiza şi evaluarea performanţelor sistemelor proiectate prin metode în domeniul frecvenţă. O metodă euristică bazată pe grafică interactivă pentru aproximarea caracteristicilor de frecvenţă a sistemelor cu dinamică complexă este prezentată în [11.2/C2]. Lucrarea [11.2/C3] propune un algoritm numeric stabil pentru calcularea numitorilor de ordin superior care intervin în studierea comportării buclelor de reacţie, individuale, ale sistemelor multivariabile. O metodologie de proiectare în domeniul timp a sistemelor cu nelinearităţi în bucla de reacţie este propusă în [11.2/C4], iar în [11.2/C5] se prezintă o abordare prin optimizare multicriterială a proiectării regulatoarelor lineare. Tehnici de analiză structurală a unor proprietăţi de bază ale sistemelor lineare sînt propuse în lucrarea [11.2/C6]. Vom trece în revistă pe scurt toate aceste metode şi tehnici în cele ce urmează.

Cînd se utilizează un pachet de proiectare asistată de calculator, proiectantul comunică cu calculatorul prin intermediul unei interfeţe, ce-i permite interpretarea rezultatelor furnizate de calculator şi specificarea operaţiilor pe care el doreşte să le efectueze. Orice informaţie pre-

zentată de calculator proiectantului, care-i permite evaluarea cantitativă a unui aspect relevant al procesului de proiectare, poartă numele de *indicator*. Astfel de indicatori sînt intim legaţi de modul de lucru conceptual al proiectantului, mod de lucru care, la rîndul său, trebuie să fie strîns legat de posibilităţile de calcul prevăzute pentru efectuarea procedurilor de sinteză cu calculatorul. Pe baza indicatorilor, proiectantul va trebui să fie în măsură să identifice sursele dificultăţilor care apar în rezolvarea problemei de proiectare şi va trebui să facă compromisurile cele mai favorabile între performanţele dorite ale sistemului automat şi costul necesar pentru realizarea acestor performanţe.

În *lucrarea 11.2/C1* se propun o serie de indicatori pentru unele caracteristici de bază ale unui sistem multivariabil, cum sînt : stabilitatea, performanţa, robusteţea sau sensibilitatea sistemului. Indicatorii propuşi reprezintă generalizări netriviale ale conceptelor de *amplificare* şi *fază*, larg utilizate în teoria clasică a reglării automate.

Considerăm un sistem cu m intrări şi m ieşiri descris de o matrice de transfer G (s) ale cărei elemente sînt funcţii raţionale în variabila complexă s. Indicatorul de stabilitate a sistemului în buclă închisă se defineşte în termenii valorilor proprii ale matricei G (s), numite şi *amplificări caracteristice*, pentru valori ale lui s pe un contur Nyquist. Acest indicator apare fie sub forma *diagramelor Nyquist generalizate* obţinute prin reprezentarea locurilor valorilor proprii, fie sub forma *diagramelor Bode generalizate* obţinute prin reprezentarea modulelor şi fazelor amplificărilor caracteristice.

Pentru studierea performanţelor sistemului în buclă închisă, se utilizează ca indicatori valorile singulare sau *amplificările principale* ale operatorilor $F(s)=I+Q(s)$ şi $L(s)=I+Q^{-1}(s)$, unde Q (s) este matricea de transfer a sistemului în buclă deschisă. Robusteţea stabilităţii sistemului, ca şi sensibilitatea sistemului la perturbaţii parametrice pot fi caracterizate convenabil în termenii amplificărilor principale.

Indicatorii propuşi în lucrare servesc la fundamentarea unei metodologii de proiectare *cuasi-clasice* a sistemelor automate multivariabile. Această metodologie este adecvată transpunerii pe calculator sub forma unei proceduri interactive de proiectare asistată de calculator.

În *lucrarea 11.2/C2* se prezintă o metodă euristică bazată pe grafică interactivă pentru determinarea unor funcţii de transfer simple pentru sisteme continue. Modelul iniţial al procesului rezultă de multe ori prea complicat pentru a putea fi manipulat comod. Astfel de modele, cu factori iraţionali sau de ordine mari, rezultă frecvent în modelarea teoretică a proceselor. Scopul problemei de reducere a modelului constă în determinarea unei aproximări suficient de exacte care să surprindă proprietăţile esenţiale ale sistemului. Modelele iniţiale acceptate de metoda propusă pot avea drept componente funcţii de transfer iraţionale, funcţii de transfer raţionale cu ordin ridicat al numitorului. De asemenea, o descriere neparametrică prin valori discrete ale răspunsului în frecvenţă este acceptată.

Reducerea modelului se bazează pe reprezentarea grafică a diagramelor Bode : amplificare-frecvenţă şi fază-frecvenţă, într-un domeniu de frecvenţe fixat, pentru modelul iniţial al sistemului. Modelul redus se obţine sub forma

$$G(s)=\frac{v}{s^r}\,\frac{\prod_i\left(1+\frac{s}{z_i}\right)\prod_j\left(1+\frac{2\zeta_j}{\omega_j}s+\frac{s^2}{\omega_j^2}\right)}{\prod_m\left(1+\frac{s}{p_m}\right)\prod_n\left(1+\frac{2\tilde{\zeta}_n}{\tilde{\omega}_n}s+\frac{s^2}{\tilde{\omega}_n^2}\right)}\,e^{-Ts} \qquad (1)$$

Procedura de reducere porneşte cu aproximarea asimptotică a diagramei amplificare-frecvenţă cu drepte de pantă k ·20 dB/decadă (k număr întreg) utilizînd facilităţi de grafică interactivă. Proiectantul efectuează această etapă prin fixarea frecvenţelor de tăiere sau a coeficienţilor de amortizare ζ şi a frecvenţelor caracteristice ce intervin în termenii pătratici din (1). Decizia proiectantului se validează la fiecare pas de către calculator prin reprezentarea suprapusă a graficului aproximativ rezultat. Printr-o procedură de încercare-eroare se potrivesc valorile caracteristice astfel încît în final să rezulte o aproximare cît mai bună a caracteristicii iniţiale. Etapa a doua constă în adăugarea unor factori suplimentari, de regulă întîrzieri, pentru găsirea unei aproximări cît mai bune şi pentru caracteristica fază-frecvenţă. Metoda se dovedeşte utilă în multe cazuri practice şi se poate utiliza uşor de proiectanţii familiarizaţi cu utilizarea diagramelor Bode.

În *lucrarea 11.2/C3* se prezintă un algoritm numeric stabil pentru calcularea *numitorilor de ordin superior* (NOS). Aceşti numitori intervin în studierea comportării buclă cu buclă a sistemelor multivariabile cu reacţie proporţională de la ieşire. NOS se definesc ca determinanţi obţinuţi prin expandarea polinomului caracteristic al unui sistem în buclă închisă în termenii coeficienţilor matricei de reacţie. Ei constituie generalizări ale polinomului de la numitor şi se demonstrează că reprezintă polinomul caracteristic al sistemului pentru amplificări infinite. Prin urmare, ei se pot utiliza în examinarea buclelor de reacţie individuale cînd există amplificări infinite în celelalte bucle.

Fie G (s) matricea de transfer a unui sistem cu m intrări şi m ieşiri şi fie factorizarea sa sub forma de produse de matrici polinomiale

$$G(s)=L^{-1}(s)\,H(s)$$

sau

$$L(s)\,y=H(s)\,u$$

unde u şi y sînt intrările şi ieşirile sistemului respectiv. Dacă legea de comandă are forma

$$u=-Ky$$

atunci polinomul caracteristic al sistemului în buclă închisă este

$$\chi(s)=\det\,[L(s)+H(s)\,K]$$

În cazul general există m^2 bucle de reacţie. Însă pentru introducerea noţiunii de NOS este suficient să considerăm K de formă diagonală

$$K=\mathrm{diag}\,[k_{ii}]$$

unde $k_{ii}\neq 0$, $i=1,\ldots p$ şi $k_{ii}=0$, $i=p+1,\ldots,m$. Atunci polinomul caracteristic al sistemului se poate scrie într-o formă expandată în termenii coeficienţilor k_{ii} în modul următor

$$\chi(s)=\Delta(s)+\sum_{i=1}^{p}k_{ii}N_{u_i}^{y_i}+\sum_{j=2}^{p}\sum_{i=1}^{j-1}k_{ii}k_{jj}N_{u_iu_j}^{y_iy_j}+\ldots+\left(\prod_{i=1}^{p}k_{ii}\right)N_{u_1\ldots u_p}^{y_1\ldots y_p}$$

Coeficienţii $N_{u_i\ldots u_j}^{y_i\ldots y_j}$ se definesc ca numitori de ordin superior şi se pot exprima ca determinanţi ai unor matrici formate din coloanele matricilor L (s) şi H (s). În expresia de mai sus, Δ (s) este polinomul caracteristic al sistemului în buclă deschisă.

Metoda de calcul propusă în lucrare, spre deosebire de metodele existente, evită evaluarea determinanţilor. NOS rezultă ca produsul polilor şi zerourilor unor sisteme de ordin redus, sisteme obţinute utilizînd algoritmul numeric stabil de determinare a structurii spectrale a unui sistem multivariabil propus de Van Dooren. Metoda preconizată pune în evidenţă conexiunea care există între NOS şi zerourile de transmisie ale sistemului.

Lucrarea 11.2/C4 descrie aplicarea în proiectarea sistemelor automate multivariabile nelineare a unor margini funcţionale ale răspunsurilor în timp ale erorilor rezultate prin linearizări. Structura sistemului este prezentată în fig. 2.

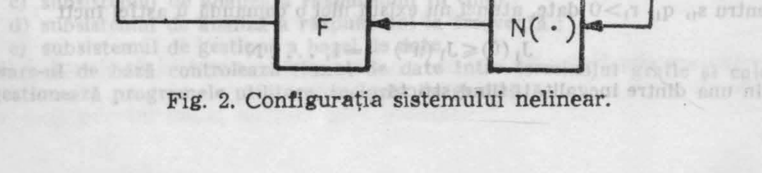


Fig. 2. Configuraţia sistemului nelinear.

Componentele sistemului includ elementul de pe legătura directă cu m-intrări şi m-ieşiri care poate combina modelul G al procesului şi compensatorul K. Pe bucla de reacţie găsim o reacţie lineară proporţională F şi o nelinearitate statică de măsurare $N(\cdot)$. Se presupune că fiecare componentă $N_{ij}(\cdot)$ se poate descompune în forma

$$N_{ij}(x)=L_{ij}x+E_{ij}(x)+V_{ij}(x)$$

unde L_{ij} este o amplificare lineară pură, $E_{ij}(\cdot)$ este o funcţie mărginită de un sector satisfăcînd

$$|E_{ij}(x)|\leqslant \bar{E}_{ij}|x|,\quad \forall x\in R$$

iar $V_{ij}(\cdot)$ este o funcţie nelineară absolut mărginită satisfăcînd

$$|V_{ij}(x)|\leqslant \bar{V}_{ij},\ \forall x\in R$$

Fie $y_i(t)$ şi $y_{Li}(t)$, $i=1,\ldots,m$ componentele ieşirilor sistemului iniţial (nelinear) şi ale sistemului linearizat obţinut pentru $E_{ij}=0$ şi $V_{ij}=0$. În lucrare se deduc, utilizînd concepte ale analizei funcţionale, un set de funcţii crescătoare $\Phi_i(t)$, $i=1,\ldots,m$ care satisfac relaţiile

$$y_{Li}(t)-\Phi_i(t)\leqslant y_i(t)\leqslant y_{Li}(t)+\Phi_i(t)$$

Această relaţie are o interpretare grafică simplă, reprezentînd o pereche de curbe echidistante faţă de $y_{Li}(t)$ între care este situat răspunsul sistemului nelinear $y_i(t)$.

O proprietate importantă a marginilor $\Phi_i(t)$ este că depind numai de parametrii sistemului linearizat. În acest fel, proiectarea sistemului se poate face utilizînd procedurile bine puse la punct ale teoriei lineare ale sistemelor, urmînd ca prin evaluarea marginilor $\Phi_i(t)$ să se verifice performanţa sistemului nelinear. În lucrare se prezintă o procedură interactivă de proiectare asistată de calculator bazată pe această idee. Proiectarea se face în domeniul timp şi se poate efectua atît în cazul sistemelor continue, cît şi discrete.

În *lucrarea 11.2/C5* se prezintă o abordare a proiectării regulatoarelor optimale pentru sisteme lineare constante prin tehnici de optimizare multicriteriale. Considerăm sistemul

$$\dot{x}=Ax+Bu,\quad x(t_0)=x_0$$

unde $x(t)\in R^n$ este vectorul de stare, $u(t)\in R^m$ este vectorul de comandă şi $t\in[t_0, t_f]$. Criteriul de performanţă care trebuie minimizat se alege de forma

$$I=x^T(t_f)Sx(t_f)+\int_{t_0}^{t_f}[x^T(t)Qx(t)+u^T(t)Ru(t)]dt$$

în care matricile S, Q şi R se presupun diagonale, cu R pozitiv definită, iar S şi Q pozitiv semidefinite. Alternativ, putem considera următoarea formulare a problemei de optimizare cu $N=2n+m$ criterii

$$J_i=\min x_i^2(t_f),\quad J_{n+j}=\min\int_{t_0}^{t_f}x_j^2\,dt,\quad J_{2n+k}=\min\int_{t_0}^{t_f}u_k^2\,dt$$

pentru $i=1,\ldots,n$; $j=1,\ldots,n$; $k=1,\ldots,m$.

Fie vectorul ponderilor

$$w^T=(s_1,\ldots,s_n,q_1,\ldots,q_n,r_1,\ldots,r_m)$$

format din elementele diagonale ale matricelor S, Q şi R, respectiv. Din teoria optimizării multicriteriale avem că toate comenzile optimale $u(t)$ rezultate prin minimizarea criteriului $I=w^TJ$, $(J=(J_1,\ldots,J_N))$ cu $w_i\geqslant 0$ sînt „soluţii neinferioare" ale problemei multicriteriale. Cu alte cuvinte, dacă u^* este soluţia optimală care rezolvă problema de minimizare iniţială cu criteriul pătratic pentru s_i, q_i, $r_i>0$ date, atunci nu există nici o comandă $\hat{u}$ astfel încît

$$J_i(\hat{u})\leqslant J_i(u^*)\quad i=1,\ldots,N$$

cu cel puţin una dintre inegalităţi fiind strictă.

Datorită convexității problemei multicriteriale, toate soluțiile neinferioare se pot obține prin varierea componentelor lui w în mulțimea $\Omega = \{w_i \mid w_i > 0\}$. Problema care se ridică este alegerea soluției convenabile din mulțimea legilor de reglare neinferioare utilizînd, de exemplu, o procedură de evaluare interactivă prin simulări repetate. Aceste legi de reglare se determină prin calcularea soluției unei probleme de optimizare linear-pătratică de dimensiune n.

Problema determinării comenzii care simultan optimizează criteriul de performanță pătratic și minimizează sensibilitate sistemului în buclă închisă în raport cu perturbații parametrice se formulează ca o problemă de minimizare bicriterială. Rezolvarea acestei probleme presupune extinderea sistemului de ecuații diferențiale și rezolvarea unei probleme de optimizare linear-pătratică de ordin 2n.

Procedurile de rezolvare s-au implementat sub forma unui pachet de programe interactiv. Pe lîngă modulele de calcul, pachetul dispune de un modul de interacțiune, care implementează o procedură de interferență lingvistică bazată pe utilizarea algoritmilor relaționali „fuzzy". Această procedură evaluează, pe baza informațiilor lingvistice primite de la utilizator, modificările care se impun în componentele vectorului de ponderi w în vederea reluării calculelor de optimizare. Programele sînt scrise în limbajul PASCAL MT+ pe un microcalculator cu procesor cu 8 biți și lucrează sub sistemul de operare CP/M.

Lucrarea 11.2/C6 prezintă utilizarea tehnicilor de manipulare algebrică în testarea unor proprietăți structurale ale sistemelor lineare descrise de ecuații de stare.

Procedurile propuse pentru testarea controlabilității, observabilităii, identificabilității etc. utilizează informațiile referitoare la structura de zerouri fixate în matricile sistemului. O parte dintre algoritmii propuși implică rezolvarea unui set de ecuații polinomiale cu mai multe necunoscute. Metoda de bază pentru aceasta se bazează pe teoria eliminării. Utilizarea acestor tehnici necesită, de regulă, un mare volum de calcul și se recomandă numai pentru sisteme de dimensiuni relativ mici.

PROIECTAREA ASISTATĂ DE CALCULATOR A SISTEMELOR CU EȘANTIONARE

Proiectarea asistată de calculator a sistemelor cu eșantionare prezintă un interes deosebit în contextul folosirii din ce în ce mai intense a structurilor de conducere cu microprocesoare a proceselor tehnologice.

În *lucrarea 11.2/B1* este prezentat un sistem de simulare interactiv ISAC-D (Interactive Simulation System for Automatic Control-Digital) destinat analizei și sintezei sistemelor de conducere numerică. Sistemul de programe este realizat folosind conceptul de diagrame-bloc introduse prin dialogul utilizatorului cu un terminal grafic inteligent (HITAC-G760) legat la un calculator mai puternic (HITAC-M200H). Este introdus un concept de organizare ierarhică a descrierii diagramelor-bloc pe 3 nivele, ceea ce permite descrierea celor mai complexe scheme de conducere. Sînt furnizate blocuri predefinite atît pentru părțile analogice, cît și pentru cele numerice ale schemelor de conducere. De asemenea, sistemul de programe conține și funcții pentru calculul răspunsului în timp al sistemului dinamic hibrid, răspunsul la frecvență, locul rădăcinilor.

În cadrul configurației descrise, terminalului grafic îi revin sarcinile de pre- și postprocesare date și de dialog om-mașină, în vederea minimizării încărcării calculatorului mare. Un meniu de funcții disponibile este oferit utilizatorului în partea de preprocesare în care se introduc datele și se descrie modelul sistemului hibrid (continuu-numeric). Partea de postprocesare oferă funcții grafice, cum ar fi rotații, extinderi și reîmprospătări de imagini.

Sistemul de programe are următoarele componente :

a) software de bază ;
b) subsistemul de introducere a diagramelor-bloc ;
c) subsistemul de analiză a răspunsului în timp ;
d) subsistemul de analiză a răspunsului la frecvență ;
e) subsistemul de gestiune a bazei de date.

Software-ul de bază controlează fluxul de date între terminalul grafic și calculatorul gazdă și gestionează programele utilitare, inclusiv de grafică.

Subsistemul de gestiune a bazei de date a fost special dezvoltat şi tratează o bază de date simplă, cu structură arborescentă.

Subsistemul de intrare a diagramelor-bloc permite construirea acestora pe terminalul video funcţie de comenzile şi structura descrisă de utilizator.

Se pot folosi elemente dinamice, algebrice lineare, algebrice nelineare, logice, elemente de întîrziere, convertoare A/N şi N/A. Pentru a simula aplicaţii mari se permite introducerea a cel mult 30 diagrame-bloc şi cel mult 500 de elemente pentru descrierea unui sistem de conducere numerică.

Întrucît pentru sistemele de conducere de mari dimensiuni afişarea lor este greoaie, necesitînd multe pagini de display, este dificil pentru utilizator să coreleze legăturile între pagini. Pot apare erori de construcţie şi descriere a diagramelor de dimensiuni mai mari. Pentru evitarea lor s-a introdus conceptul de descriere structurată a diagramelor-bloc. Diagramele-bloc structurate sînt clasificate după cum urmează :

(i) *nivelul overview* (general), care descrie întreg sistemul pe subsisteme componente ;
(ii) *nivelul detaliu*, care poate conţine pînă la 28 de pagini şi conţine componentele fiecărui subsistem ; fiecare pagină se conectează cu cele adiacente, prin intrări/ieşiri ;
(iii) *nivelul rutină* (pagină), pe o singură pagină de ecran se grupează elemente care se consideră o singură funcţie de transfer la nivelul doi.

Separarea părţii continue de cea discretă în descriere se face cu ajutorul blocurilor A/N, N/A cu cîte 6 intrări/ieşiri fiecare. Pe lîngă frecvenţa de eşantionare, blocurile sînt prevăzute şi cu timp de calcul necesar algoritmului de reglare, eroare de trunchiere etc., făcînd astfel ca aceste canale de trecere a informaţiei să se apropie foarte mult de implementarea lor reală, cu microprocesoare.

Pentru a obţine rezultate numeric stabile în procesul de simulare a funcţionării sistemului de conducere, autorii au ales o metodă de discretizare a părţii continue (nelineară în general), cu ajutorul metodei tabloului, după cum urmează :

$$\dot{x}_1(t) = a_{1j}x_1(t) + a_{2i}x_{\alpha_1}(t) + a_{3i}x_{\beta_1}(t) \qquad (1)$$

unde α_1, β_1 sînt indici pentru mărimile de intrare ;

$$x_j(t) = a_{2j}x_{\alpha_2}(t) + a_{3j}x_{\beta_2}(t) + a_{4j}x_{\gamma_2}(t) + \delta_j \qquad (2)$$

unde δ_j este intrarea în proces

$$x_1(t) = f_1(x_{\alpha_3}(t),\ x_{\beta_3}(t)) \qquad (3)$$

Aplicînd o formulă de discretizare într-un pas, ecuaţia (1) devine :

$$[1-h(1-\theta)]\,x_1(k+1) - h(1-\theta)\,a_{2i}x_{\alpha_1}(k+1) - h(1-\theta)\,a_{3i}x_{\beta_1}(k+1) =$$

$$= (1 + h\theta a_{1i})\,x_1(k) + h\theta a_{2i}x_{\alpha_1}(k) + h\theta a_{3i}x_{\beta_1}(k) \qquad (4)$$

unde θ este o pondere între 0 şi 1.

Intervalul $[0, T]$ se divide în $0, t_1, \ldots, t_n$ puncte.

Pentru ecuaţia (3) se aplică iteraţii de tip Newton-Raphson pornind de la valorile $x_1(k)$, $x_{\alpha_3}(k)$ şi $x_{\beta_3}(k)$

Obţinem :

$$x_1(k+1) - \left(\frac{\partial f_1}{\partial x_{\alpha_3}}\right) x_{\alpha_3}(k+1) - \left(\frac{\partial f_1}{\partial x_{\beta_3}}\right) x_{\beta_3}(k+1) =$$

$$= x_1(k) - \left(\frac{\partial f_1}{\partial x_{\alpha_3}}\right) x_{\alpha_3}(k) - \left(\frac{\partial f_1}{\partial x_{\beta_3}}\right) x_{\beta_3}(k). \qquad (5)$$

Rezultă astfel tabloul valorilor discrete

$$A(k)\cdot X(k+1) = B(k) \qquad (6)$$

unde X este un vector N-dimensional $(x_1, \ldots, x_N)$ cu N — numărul de blocuri, $A(k) \in R^{N \times N}$, $B(k) \in R^N$.

Pentru partea discretă este furnizată o tehnică de calcul secvenţial bazată pe observaţia că un algoritm numeric de reglare este un model discret în timp. Deci o diagramă-bloc a unui regulator numeric poate fi transformată în ecuaţii cu întîrzieri (cu unitatea de timp dată de pasul de calcul), în ecuaţii algebrice lineare şi nelineare şi în ecuaţii logice. Secvenţa porneşte de la valorile iniţiale pe ecuaţiile cu întîrzieri, valorile intrărilor şi ieşirilor din convertoarele A/N şi N/A. Comunicaţia între cele 2 părţi analogice şi numerice se face prin intermediul intrărilor şi ieşirilor acestor convertoare.

O altă componentă a sistemului descris se referă la calculul răspunsului la frecvenţă a sistemului de reglare, care-şi găseşte utilitatea pentru sisteme complexe de reglare, pentru care nu se poate calcula atît de uşor funcţia de transfer a acestuia. Metoda pleacă de la sistemul discret :

$$(zI - A_D)\, X_D(z) = D_D\, U(z) \tag{7}$$

$$Y(z) = C_D X_D(z) \tag{8}$$

şi rezolvă o ecuaţie algebrică de tipul :

$$|\exp(j\omega T_s)\, I - A_D|\, X_D(\omega) = (D_D)_i \tag{9}$$

unde : T_s este perioada de eşantionare ; matricile A_D, $(D_D)_i$ sînt matricile sistemului dinamic discretizat, X_D — starea sistemului dinamic discret, $(D_D)_i$ — e coloana (i) a matricii (D_D). Din (9) se obţine :

$$Y_i = (C_D)_i\, (X_D)_R + (C_D)_i\, (X_D)_I = Y_R + Y_I \tag{10}$$

unde : $(C_D)_i$ este coloana i a matricii C_D ;

R, I — atributul real sau imaginar al părţii lui X_D.

Se obţin astfel uşor diagramele Bode pentru întreg sistemul de conducere. În plus, folosind tehnici QR se trasează şi poziţia rădăcinilor sistemului, obţinîndu-se o analiză a stabilităţii sistemului cînd acesta este liniarizat şi discretizat (ec. 7).

În *lucrarea 11.2/B2*, autorii prezintă o metodă exactă pentru evaluarea ciclurilor limită a frecvenţe subarmonice în sistem numeric nelinear, obţinută prin modificarea metodei pentru sistemele continue cu nelinearitate de tip releu. Metoda se poate aplica la orice tip de nelinearitate, spre deosebire de cazul continuu.

Metoda propusă implică găsirea soluţiilor unor ecuaţii algebrice nelineare şi presupune existenţa unei estimări iniţiale a soluţiilor. Pulsurile funcţiei de nelinearitate au lăţimea $\Delta t_i = T_s$. Întrucît extrapolatorul de ordin zero precede elementul nelinear, ieşirea y (t) a acestuia va fi un tren de impulsuri a cărui expresie Fourier este :

$$y_i(t) = \frac{h\Delta t_i}{T} + \frac{h}{\pi}\sum_{n=1}^{\infty}\frac{1}{n}[\sin n\omega\Delta t_i \cos n\omega(t - t_i) + $$

$$+ (1 - \cos n\omega\Delta t_i) \sin n\omega(t - t_i)] \tag{11}$$

Deci ieşirea sistemului va deveni :

$$c_i(t) = \frac{h\Delta t_i G(0)}{T} + \frac{h}{\pi}\sum_{n=1}^{\infty}\frac{g_n}{n}[\sin n\omega\Delta t_i \cos(n\omega t - $$

$$- n\omega t_i + \Phi) + (1 - \cos n\omega\Delta t_i)\sin(n\omega t - n\omega t_i + \Phi)] \tag{12}$$

În continuare autorii arată că pentru a obţine condiţiile de oscilaţie, trebuie aflată amplitudinea ciclului limită la intrarea în elementul neliniear. Deci :

$$-x[(K-1)T_s] = c[(K-1)T_s] \tag{13}$$

unde : K = 1, 2, . . ., n ;

T_s este perioada de eşantionare.

Din expresiile celor 2 membrii ai ecuaţiei (13) rezultă ecuaţiile algebrice nelineare care trebuie rezolvate. De cele mai multe ori procedura iterativă de căutare a soluţiilor este convergentă chiar dacă punctul iniţial nu este bine precizat.

Pentru a estima valorile perioadelor ciclurilor limită din sistem se folosește o funcţie de descriere a ansamblului eșantionator, extrapolator de ordin zero și element nelinear. Această funcţie de descriere este valabilă pentru cicluri limită care au perioada un multiplu întreg al perioadei de eșantionare.

Funcţia de descriere evaluată prin această metodă poate fi afișată grafic ca o dreaptă ce trece prin origine și face un unghi α cu axa pe diagrama Nyquist. Locul de intersecţie al hodografului $G(j\omega)$ cu $-\frac{1}{N(a, \omega)}$ va da ciclul limită.

Programele dezvoltate pentru rezolvarea acestor probleme sînt în număr de 3 și permit estimarea ciclurilor limită în astfel de sisteme nelineare cu eșantionare și calculul soluţiilor exacte.

Astfel, programul SAMDF.F77 implementează metoda funcţiei de descriere pentru a putea estima valorile iniţiale perioadelor ciclurilor limită, care pot fi folosite în celelalte 2 programe pentru aflarea soluţiilor exacte.

Utilizatorul descrie funcţia Laplace a instalaţiei și selectează nelinearitatea dintr-o tabelă dată de program. Pe lîngă metoda funcţiei de descriere, programul mai trasează și diagramele Nyquist și Bode pentru partea lineară a sistemului. Se efectuează o căutare automată a intersecţiilor într-un domeniu dat al frecvenţei. Funcţia de grafică interactivă permite dialogul și trasarea succesivă a diagramelor, permiţîndu-se astfel obţinerea ușoară a estimărilor. Celelalte 2 programe SSAM.FTN și ASAM.FTN implementează metoda exactă de găsire a ciclurilor limită cu simetrie impară (SSAM.TFN) sau asimetrice (ASAM.FTN).

APLICAŢII ALE PROIECTĂRII ASISTATE DE CALCULATOR

Secţiunea 11.2/D prezintă aplicaţii ale proiectării asistate de calculator în chimie [11.2/D2], robotică [11.2/D3, 11.2/D4] și acţionări hidraulice [11.2/D5]. Lucrarea [11.2/D2] prezintă un sistem de programe care simulează reglarea pH-ului pentru o clasă generală de procese chimice. Sistemul asigură posibilitatea folosirii algoritmilor clasici sau adaptivi de comandă, precum și identificarea în timp real a procesului condus. Programele pot fi folosite în două moduri : a) pentru proiectarea structurilor de comandă ale unei instalaţii existente ; b) pentru analiza unei instalaţii noi din punct de vedere al comenzii.

Lucrarea [11.2/D3] prezintă o metodă de reprezentare a dinamicii structurilor mecanice de manipulare. Noţiunea de elipsoid de inerţie, folosită la caracterizarea unui singur corp rigid, este extinsă la întreaga structură de manipulare. Prin desenarea elipsoidului general de inerţie se pot pune în evidenţă proprietăţile distribuţiei de mase și caracteristicile dinamice ale întregului manipulator. Această metodă este folosită la optimizarea distribuţiei de mase și la alegerea dimensiunilor structurii mecanice de manipulare.

Lucrarea [11.2/D4] prezintă un sistem de programe pentru modelarea automată a structurilor mecanice de manipulare antropomorfice, folosit în proiectarea asistată de calculator a manipulatoarelor și roboţilor industriali. Sistemul generează automat, sub formă simbolică, modelele geometrice, cinematice și dinamice ale structurilor mecanice de manipulare. Pot fi analizate structuri mecanice complexe, cu lanţuri cinematice deschise și (sau) închise.

Lucrarea [11.2/D5] prezintă un pachet de programe pentru analiza și sinteza sistemelor de acţionare hidraulică și electrohidraulică. Este prezentată o metodă teoretică nouă de modelare matematică a componentelor și a întregului sistem de acţionare.

Lucrarea 11.2/D2 descrie un sistem de programe realizat în limbaj FORTRAN și implementat pe calculatorul VAX 11/780, pentru simularea reglării pH-ului în instalaţii chimice. Acest sistem poate fi folosit pentru instalaţii existente la determinarea performanţelor de reglare optimă, dintr-un set de algoritmi de comandă cunoscuţi de sistem, la determinarea eficienţei implementării unui anumit algoritm de reglare, la analiza modificării performanţelor cînd se schimbă condiţiile de operare.

În cazul proiectării unei noi instalaţii tehnologice, se pot determina cu ajutorul sistemului de programe condiţiile de operare, valorile parametrilor, structura instalaţiei etc. Algoritmii de comandă pe care îi poate folosi sistemul de programe variază de la clasicul PI, la algoritmi adaptivi autoacordabili sau la algoritmi stohastici de comandă suboptimală. În lucrare se pre-

zintă pe larg metoda de modelare matematică a instalațiilor tehnologice folosite de sistemul de simulare. Caracteristicile interactive ale sistemului permit proiectantului să definească, să modifice și să memoreze modelele matematice ale instalațiilor conduse, să efectueze simulări și să memoreze rezultatele de simulare. În lucrare se prezintă și aplicații tipice în care a fost folosit sistemul de programe.

Lucrarea 11.2./D3 prezintă caracteristicile structurale și dinamice ale manipulatoarelor într-o formă ușor de analizat și modificat de către proiectantul structurii mecanice. Tehnicile actuale de simulare a dinamicii roboților sînt greu de utilizat, necesitînd un volum mare de calcule pentru diferitele traiectorii și configurații ale robotului. Noua metodă prezentată este eficientă din punctul de vedere al proiectării și evaluării caracteristicilor dinamice ale structurilor mecanice de manipulare.

Extinderea noțiunii de elipsoid de inerție la întreaga structură și folosirea unui sistem grafic puternic reprezintă caracteristicile principale ale sistemului prezentat în lucrare. Elipsoidul de inerție este asociat cu energia cinetică înmagazinată de corpul rigid. Pentru definirea elipsoidului de inerție generalizat asociat întregii structuri, se calculează energia cinetică totală a manipulatorului. Aplicînd una din metodele matriciale de calcul a energiei cinetice totale, pentru structura mecanică de manipulare, rezultă:

$$T = 1/2\dot{q}^t\, G(q)\, \dot{q}$$

unde T este energia cinetică, $\dot{q}$, $\dot{q}^t$ — vectorul și vectorul transpus al vitezelor generalizate, G (q) — matricea energiei cinetice, care se definește ca tensorul generalizat de inerție pentru o structură de corpuri rigide. Tensorul generalizat de inerție are o interpretare geometrică, care pune sugestiv în evidență caracteristicile dinamice ale manipulatorului. Se asociază tensorului generalizat de inerție suprafața:

$$u^t\, G(q)\, u = 1$$

care reprezintă ecuația unui elipsoid, denumit elipsoidul generalizat de inerție (EGI). Axele principale de inerție ale elipsoidului (EGI) sînt colineare cu vectorii proprii ai matricii G, iar lungimea axelor este funcție de rădăcina pătrată a valorilor proprii.

Deoarece este interesantă dinamica mișcării mîinii robotului în coordonate carteziene, se calculează matricea G în coordonate carteziene și se desenează mai mulți elipsoizi (EGI) în spațiul de lucru al robotului. Formele grafice ale acestor elipsoizi trasate în diferite poziții ale mîinii robotului, în spațiul de lucru, pun în evidență caracteristicile dinamice ale structurii mecanice, funcție de poziția și direcția de mișcare a mîinii. În lucrare se extinde și noțiunea de moment de inerție a unui corp rigid, la momentul de inerție generalizat a unei structuri mecanice de manipulare:

$$h = \dot{q}^t G \dot{q} / \dot{q}^t \dot{q}$$

Momentul de inerție generalizat depinde de poziția în spațiul de lucru și de direcția de mișcare a mîinii robotului. Valoarea maximă (minimă) a lui h corespunde la valorile proprii maxime (minime) ale matricii G, care corespund la axele mici (mari) ale elipsoidului generalizat de inerție. Viteza de mișcare a structurii (raportată la mîna robotului) este minimă în direcția axelor mici ale elipsoidului (EGI). Rezultă deci că din interpretarea formelor grafice (EGI) desenate pentru spațiul de lucru, se pot trage concluzii asupra posibilităților de mișcare ale robotului, asupra punctelor singulare, asupra gradului de singularitate etc.

Forma elipsoidului (EGI), deformarea și rotirea lui sînt legate de valorile forțelor Coriolis și centrifugale ale structurii mecanice. În lucrare se demonstrează că pentru zonele din spațiul de lucru, în care elipsoidul (EGI) se transformă într-o sferă, forțele nelineare Coriolis și centrifugale dispar. Efectele inerțiale, variația forțelor Coriolis și centrifugale, în spațiul de lucru, interpretate prin reprezentările grafice ale elipsoidului generalizat de inerție, sînt tot atîtea date utile pentru proiectantul structurii mecanice. În faza de proiectare se modifică dimensiunile geometrice și distribuția de mase, astfel ca elipsoidele de inerție să tindă, în spațiul de lucru, cît mai mult spre sfere. În acest fel se asigură bune proprietăți dinamice pentru structura mecanică de manipulare.

Lucrarea 11.2.D4 prezintă modelarea automată a structurilor mecanice complexe cu lanțuri cinematice deschise și (sau) închise. Pentru automatizarea modelării structurilor mecanice cu lanțuri cinematice închise se asociază manipulatorului un graf orientat (corp rigid— vîrf de graf ; legătură cinematică-arc de graf). În etapa de analiză topologică se determina drumul minim în graf, corespunzător lanțului fundamental de poziționare (succesiunea de corpuri rigide care unesc mîna robotului de batiul său) și o bază de cicluri minimale independente, corespunzătoare grupelor minimale (lanțuri cinematice închise caracterizate de o succesiune predeterminată de corpuri, un corp de bază, un operator global de mișcare, o ieșire și una sau mai multe intrări).

Sistemul de programe folosește un fișier cu grupe minimale cunoscute, descrise prin relațiile analitice ale operatorilor globali de mișcare. În prima fază a analizei topologice se pune în evidență lanțul fundamental de poziționare și grupele minimale ale structurii particulare, identificîndu-se automat formele analitice ale operatorilor globali de mișcare. Dacă nu toate legăturile motoare se află în lanțul fundamental de poziționare, atunci se construiesc automat lanțurile de acționare (succesiunea de corpuri care unesc legăturile motoare cu o legătură din lanțul fundamental de poziționare).

Folosind operatorii globali de mișcare ai grupelor minimale, se generează automat, în formă simbolică, pentru fiecare corp rigid al robotului, vectorul de poziție al originii reperului de coordonate legat de corp, vectorul de poziție al centrului de masă al corpului și matricea de rotație față de sistemul de referință inerțial. Elementele acestor vectori și matrici au ca variabile coordonatele generalizate ale robotului ($q \in R^n$, n — numărul gradelor de libertate). Vectorul și matricea simbolică corespunzătoare mîinii robotului reprezintă modelul geometric al întregii structuri mecanice de manipulare.

În faza de analiză cinematică se generează automat, folosind derivarea simbolică, matricile de transmitere de ordinul întîi, corespunzătoare vitezei lineare și unghiulare a fiecărui corp, față de sistemul de referință inerțial. Matricile de transmitere corespunzătoare mîinii robotului formează matricea Jacobian a modelului cinematic.

Cu ajutorul vitezelor unghiulare, a vitezelor lineare ale centrelor de greutate, a maselor și momentelor de inerție ale corpurilor rigide, se calculează energia cinetică a întregii structuri. Folosind rezultatele analizei cinematice și poziționale, aplicate tuturor corpurilor, se calculează și energia potențială a manipulatorului. Cu ajutorul ecuațiilor lui Lagrange, se determină ecuațiile diferențiale de mișcare, respectiv modelul dinamic al structurii mecanice de manipulare. Forma analitică a modelului dinamic rezultat este :

$$A(q)\,\bar{q} + B(q)\,\dot{q}\dot{q} + C(q)\,\dot{q}^2 = Q_g(q) + Q_m + Q_r(q)$$

unde $A(q) \in R^{n\times n}$ este matricea energiei cinetice sau matricea forțelor de inerție, $B(q) \in R^{n\times n(n-1)/2}$ — matricea forțelor Coriolis, $C(q) \in R^{n\times n}$ — matricea forțelor centrifugale, $Q_r(q) \in R^n$ — vectorul forțelor de rezistență, $Q_m \in R^n$ — vectorul forțelor motoare, $Q_g \in R^n$ — vectorul greutății proprii, q, $\dot{q}$, $\ddot{q}$ — coordonatele, vitezele și accelerațiile generalizate, n — numărul gradelor de libertate.

Sistemul de programe este scris în limbajele PASCAL, FORTRAN, MACRO-11 și funcționează pe minicalculatoare de tip DEC PDP-11. Acest sistem cuprinde următoarele funcții :
— descrierea și modificarea grupelor minimale standard ;
— descrierea topologică a manipulatorului sau a robotului industrial ;
— analiza topologică și pozițională ;
— analiza cinematică și dinamică ;
— calculul simbolic general la nivel de matrici (+, —, ×, inversare, derivare etc.) ;
— generarea automată a programelor în limbaj FORTRAN, care descriu modelul dinamic în vederea simulării ;
— simularea numerică ;
— reprezentarea grafică a rezultatelor.

Evident, structurile mecanice formate din lanțuri cinematice deschise sînt un caz particular a structurilor mecanice cu lanțuri cinematice închise.
În încheierea lucrării se prezintă două exemple de modelare, o structură cu două grade de libertate și lanțuri cinematice închise și o structură cu trei grade de libertate și lanțuri cinematice deschise.

Lucrarea 11.2/D5 prezintă un sistem de programe pentru proiectarea instalațiilor de acționare hidraulice și electrohidraulice, folosind modelarea structurată. Modelarea structurată

permite organizarea sistematică a elementelor componente, care produc, controlează şi transmit energia hidraulică şi informaţia în instalaţii. Metoda prezentată permite o bună structurare, o optimizare locală, eliminarea erorilor de modelare şi o analiză mai uşoară a instalaţiilor complexe. Modelarea efectuată ajută sinteza asistată de calculator a instalaţiilor hidraulice şi electrohidraulice. Lucrarea prezintă pe larg unităţile elementare folosite, structurile tipice de instalaţii, descrierea lor dinamică şi determinarea modelului matematic pentru analiza dinamică şi de regim staţionar. Sistemul de programe cuprinde trei părţi : preprocesor, procesor, postprocesor. Rolul preprocesorului este de a genera un model de calcul, selectînd şi asamblînd modele matematice ale modulelor elementare. Procesorul analizează în mod opţional regimul staţionar, regimul tranzitoriu, răspunsul la frecvenţă etc.

Sistemele grafice sînt folosite pentru prezentarea rezultatelor într-o formă uşor de analizat. Postprocesorul permite selectarea acelor module elementare, care fac instalaţia să funcţioneze conform parametrilor de proiectare doriţi. Sistemul foloseşte indici globali de calitate, permiţînd utilizarea şi de criterii economice. Postprocesorul furnizează operatorului o listă de elemente modulare alese, asociată cu o listă a performanţelor obţinute de instalaţia construită din elementele alese. Comunicarea cu sistemul de programe, care funcţionează pe un minicalculator CORAL 4011, se poate realiza în mod interactiv. Lucrarea pune în evidenţă noi concepte şi metode de modelare şi construcţie automată a instalaţiilor de acţionare hidraulică şi electrohidraulică.

INGINERIA SISTEMELOR INDUSTRIALE

Ing. *Alexandru Dan Donciulescu*

I.T.C.I.

Sub titlul „Ingineria sistemelor industriale" au fost grupate lucrările, susţinute la al IX-lea Congres IFAC, care prezintă tehnici, algoritmi, structuri şi metode de proiectare specifice sistemelor de conducere cu calculator a proceselor industriale complexe.

Prima dintre cele două *secţiuni* (*11.3/A*) ale colocviului cu titlul de mai sus a conţinut 6 lucrări (dintre care lucrarea 11.3./A4 nu este cuprinsă în volumul lucrărilor congresului) axate pe conducerea unităţilor industriale mari din domeniul metalurgiei, industriei sticlei şi energeticii, precum şi pe prezentarea unor metode de analiză, sinteză şi evaluare a costului pentru sistemele complexe de conducere din domeniul industrial. A doua *secţiune* (*11.3/B*) a cuprins 3 lucrări referitoare la conducerea cu calculator a unor activităţi din industria extractivă şi de prelucrare a minereurilor.

Nefiind profund diferenţiate, prezentarea comunicărilor cuprinse în volumul lucrărilor Congresului în secţiunile 11.3/A şi 11.3/B se va face împreună. În cele ce urmează vor fi succint prezentate întîi lucrările 11.3/A1, 11.3/A3 şi 11.3/B1, care abordează conducerea cu calculator a unor unităţi de producţie din industria siderurgică, inclusiv a depozitelor de materie primă care le deservesc. În continuare se va prezenta abordarea problemei exploatării cîmpurilor petrolifere [11.3/B2] şi îmbunătăţirii factorului de recuperare în extracţia petrolului [11.3/B3]. Apoi se vor descrie sumar conducerea cu calculator de proces a două sisteme din industria sticlei [11.3/A2] şi dintr-un accelerator complex [11.3/A5], care vor permite abordarea problemei analizei şi sintezei sistemelor de conducere a proceselor complexe [11.3/A6]. În final se vor face cîteva referiri la o metodă de evaluare a efortului de realizare a software-ului de aplicaţie pentru sisteme complexe [11.3/A5].

Pătrunderea tot mai accentuată a dezvoltărilor tehnologice în sistemele industriale creează premisele perfecţionării corespunzătoare a organizării şi metodelor de conducere. Un exemplu de sistem integrat de conducere a activităţilor de producţie dintr-un laminor de benzi la cald (LBC) îl constituie cel de la Mizushima (Japonia), prezentat în *lucrarea 11.3/A3*. Sistemul este realizat pe mai multe niveluri de conducere. La nivelul superior, un calculator, cuplat on-line cu restul sistemului, realizează funcţii specifice conducerii producţiei. La nivelul următor există o serie de subsisteme care realizează controlul cuptoarelor, conducerea operativă şi fixarea referinţelor pentru procesele tehnologice, precum şi subsisteme care controlează scoaterea slaburilor din cuptoarele de încălzire şi plasarea lor pe linia de laminare. Un subsistem aparte asigură determinarea temperaturii de reîncălzire în cuptoare, minimizînd consumul de energie.

Subordonat direct nivelului superior există un nivel de control on-line al calităţii, care are următoarele funcţiuni: determinarea şi aprecierea dimensiunilor şi temperaturii slaburilor; estimarea proprietăţilor mecanice din datele operative şi acceptarea/rejectarea calitativă a slaburilor; detectarea anomaliilor slabului şi adaptarea sistemului de laminare (viteză caje) în funcţie de caracteristicile materialului. Bazat pe un mare număr de puncte de măsurare, se poate realiza un sistem de control on-line pe toată lungimea semifabricatului, asigurîndu-se în acelaşi timp şi optimizarea operării. Sistemul de control al calităţii utilizează metode avansate de prelucrare matematică (filtrare, medieie, nivelare etc.) a semnalelor primite de la traductoare. Cîteva dintre aceste metode sînt prezentate în lucrarea 11.3/A3, în cazul determinării automate a temperaturii, dimensiunilor, erorilor de formă, defectelor din structura metalului şi estimării unor proprietăţi mecanice ale laminatului.

Funcţionarea acestor sisteme este condiţionată de un mare volum de date, culese on-line din procesul de laminare. Aceste date sînt preluate prin intermediul unor microcalculatoare, care asigură prelucrările necesare interpretării datelor primite de la traductoarele din proces.

La combinatul siderurgic din Mizushima structura de conducere prezentată este suportată de două calculatoare universale (unul lucrînd pe loturi, altul on-line), 2 calculatoare de proces pentru funcţiile de supervizare, dintre care unul (cel pentru controlul calităţii) poate prelua la nevoie şi funcţiunile celuilalt ; funcţiile de colectare şi interpretare primară date sînt realizate de microcalculatoare de proces, prin intermediul cărora se face şi comandarea mărimilor de execuţie ale procesului controlate de sistem.

În cazul unui sistem integrat de conducere a activităţii de producţie dintr-un combinat siderurgic subsistemele de mai sus aparţin conducerii de proces. Ele sînt subordonate subsistemelor de conducere a producţiei, care pot asigura mărimile de referinţă necesare proceselor de producţie pentru funcţionarea eficientă a întregului sistem, conform unor obiective globale.

Programarea producţiei într-un laminor, funcţiunea cea mai importantă de conducere a producţiei, implică realizarea unui compromis între diferite criterii de eficienţă (cîteva mai practice : productivitatea laminorului, calitatea produsului final — urmărită, de exemplu, în lucrarea 11.3/A3 — consumul de energie, funcţionarea uniformă). În general programarea optimă a producţiei trebuie considerată iniţial ca o problemă de optimizare multicriterială. Rezolvarea acestei probleme prin reducerea la un singur criteriu (fie prin comasarea ponderată a funcţiilor obiectiv, fie prin transformarea criteriilor suplimentare în restricţii) nu dă în toate situaţiile soluţii acceptabile pentru toate criteriile, uneori rezultatul fiind chiar necorespunzător.

Programarea producţiei unui laminor se face în *lucrarea 11.3/A1* printr-o metodă de programare dinamică multicriterială. Dintre cele trei criterii de optim considerate importante (consum de energie, productivitate, forma semifabricatului) au fost reţinute primele două, al 3-lea fiind tratat ca o restricţie. Din soluţiile obţinute ca optim global în sens Pareto, operatorul poate selecta, în funcţie de diferite preferinţe de moment, varianta care se va implementa. Comparativ cu programele de producţie empirice, obţinute în sistemul manual, rezultatele obţinute îmbunătăţesc remarcabil funcţionarea laminorului (dublarea productivităţii, reducerea consumului de energie etc.).

Abordarea amintită consideră procesul de laminare ca un proces de decizie în mai multe etape (N), corespunzătoare numărului de treceri ale metalului prin caje. Variabila de stare la pasul k, x^k, este grosimea de intrare a semifabricatului la pasul respectiv. Variabila de decizie la acest pas este un vector u^k cu patru componente (ajustarea secţiunii laminatului — d^k, viteza de laminare şi forţele de împingere înainte, respectiv înapoi). Ecuaţia de stare este deci

$$x^{k+1}=x^k-d^k$$

Există un număr însemnat de restricţii impuse de forma laminatului, forţa de laminare, puterea motoarelor, viteza de laminare etc. Criteriul este un vector de două componente : consumul de energie şi timpul total de prelucrare (care este reciproca productivităţii). Pentru un model de programare dinamică multicriterială care defineşte setul de puncte optime în sens Pareto pentru programele de laminare se foloseşte o metodă derivată din cele cunoscute în programarea dinamică convenţională. Soluţia, respectiv programul de laminare, indică valorile variabilelor de decizie pentru fiecare trecere ; ea se obţine într-un număr de iteraţii egal cu numărul de treceri pentru laminare N, astfel că din mulţimea de soluţii fezabile se reţin cele acceptabile practic, pentru care $N \leqslant 5$. Pe baza experienţei sale sau a unor cerinţe de moment, operatorul selectează o variantă, înclinînd balanţa către unul sau altul dintre cele două criterii.

Calculatorul a pătruns în majoritatea proceselor din industria siderurgică (fabricarea oţelului şi fontei, laminare). O zonă mai puţin abordată, care oferă însă un potenţial notabil de creştere a eficienţei, este controlul calităţii şi programarea tratării minereului pentru pregătirea materiei prime. Diferitele depozite lucrează de obicei independent, pe baza unor programe de producţie empirice. Operaţiile de prelucrare dintr-un depozit sînt variate şi sînt greu de condus în cazul tratării lor interdependente, afectate de perturbaţii.

În combinatul de la Kagogawa (Japonia), conform *lucrării 11.3/B1*, atît datele necesare, cît şi programele de producţie specifice depozitelor de materii prime din prelucrarea minereului de fier sînt furnizate de calculator. Obiectivele sistemului sînt reducerea costului prelucrării minereului şi menţinerea stabilităţii în procesele de producţie din aval, printr-o alimentare corespunzătoare cu materie primă.

Sistemul de conducere a depozitelor este realizat pe mai multe nivele. Nivelul superior este constituit de un sistem de conducere pe termen lung a aprovizionării cu minereu şi de planificare a mijloacelor de transport. Planul anual de aprovizionare este defalcat pentru perioade

mai scurte de timp utilizînd programarea liniară pentru sisteme mari. Planul rezultat indică diferitele calități de minereu cu care trebuie alimentate furnalele, uzinele de sinterizare și de peletizare, precum și proporțiile în care trebuie amestecate aceste calități pentru asigurarea materiei prime. Al doilea nivel constă din sistemul de conducere a aprovizionării și formulare a cererii. Al treilea nivel asigură conducerea operațiilor din depozit și este în legătură directă cu diferitele procese de producție specifice prelucrării fontei și oțelului. La acest nivel funcțiunile principale sînt de programare a operațiilor specifice, în care resursele urmărite sînt spațiile de depozitare și mijloacele de transport intern.

Programarea stocării minereului presupune determinarea timpului de descărcare pentru fiecare intrare de minereu, precum și a locației atribuite pentru fiecare sortiment de minereu descărcat. Datorită numărului mare de sortimente și de intrări de aprovizionare, utilizarea unui model este dificilă. Se folosește o tehnică de simulare cu increment variabil de timp, care presupune : preluarea datelor privind situația actuală a stocurilor și programul de aprovizionare și de producție ; calcularea evenimentelor de timp care presupun mișcări în stocuri și ordonarea lor cronologică ; efectuarea pas cu pas a mișcărilor în stocuri conform evenimentelor și unor criterii de selectare a variantei preferate bazate pe experiența operatorului. Rezultatele simulării prezintă, într-o formă ușor de interpretat, situația fiecărei locații din depozit la diferitele momente de timp ale programului și permite operatorului organizarea eficientă a celorlalte activități din depozit.

Programarea activităților de sfărîmare/sortare/amestec implică stabilirea sortului și cantității de minereu, precum și a timpilor de începere/terminare a operațiilor respective pentru a asigura fără întreruperi materia primă necesară proceselor de prelucrare a fontei și oțelului. Luînd ca exemplu operația de sfărîmare a unui lot compus din diferite sorturi de minereu, pentru programare se consideră o serie de variabile reale (exemplu, timpul de început și de sfîrșit al operației pentru fiecare sort) și o serie de variabile 0—1 (reprezentînd, de exemplu, opțiunea dacă un anumit sort de minereu trebuie sfărîmat sau nu, precum și ordinea de sfărîmare). Problema de programare se formulează ca un program linear cu variabile mixte. Criteriul urmărește maximizarea cantității totale de minereu sfărîmat în cadrul lotului în condițiile respectării unor restricții care asigură :

— un stoc suficient de materie primă ;
— prelucrarea lotului printr-o singură operație ;
— respectarea termenelor reparațiilor mijloacelor de transport intern ;
— asigurarea spațiului de depozitare pentru minereul brut care urmează să sosească.

De o mare importanță în asigurarea unui timp de răspuns satisfăcător este reducerea numărului de variabile, ceea ce presupune reducerea numărului de sorturi de minereu considerate în problema de programare, care se poate realiza de către operator pe baza experienței sale. Considerînd, de exemplu, că pentru aplicația de la combinatul Kagogawa programul operativ de sfărîmare (care a presupus circa 30 variabile reale și 50 variabile întregi), timpul de răspuns pe un calculator IBM 3033 a fost de circa 10 min, se consideră ca o valoare acceptabilă un număr de 15 sorturi de minereu.

Construite pe principiile sistemelor suport pentru decizie, subsistemele de programare a activităților din depozit permit operatorului selectarea din mulțimea soluțiilor fezabile a variantelor care corespund mai bine situațiilor particulare, alegere în care este implicată experiența operatorului.

Industria extractivă și în special cea a petrolului este de asemenea un domeniu în care și-au căutat aplicație metodele avansate de cercetare operațională. Motivul este evident, considerînd că datorită producțiilor mari în acest domeniu, îmbunătățirea metodelor de conducere chiar cu procente mici, atrage obținerea unor cîștiguri spectaculoase de materii prime deficitare.

În domeniul extracției petrolului obiectivul principal urmărit este îmbunătățirea factorului de recuperare a petrolului, respectiv a raportului dintre volumul de petrol recuperat și volumul total al bazinului exploatat. Tratarea problemei se poate face la nivelul unei unități de exploatare (ca în lucrarea 11.3/B3) sau pe ansamblul exploatărilor unui cîmp petrolifer (ca, de exemplu, în lucrarea 11.3/B2).

Modelul de exploatare a cîmpului de petrol conține : ecuațiile de filtrare pentru amestecul petrol-apă în mediul poros al cîmpului (care constituie un sistem dinamic de ecuații nelineare cu derivate parțiale care descriu distribuția de presiuni și saturația de petrol în cîmp) ; ecuațiile de mișcare a lichidului la urcarea în puțurile de extracție ; ecuațiile care descriu raportul

dintre puțurile producătoare și sistemul de exploatare ; setul de restricții tehnologice. O particularitate privind aceste modele este că ele se pot construi după o exploatare de cîțiva ani a sistemului respectiv, necesară pentru acumularea de date suficiente din activitatea geologică.

Variabilele de comandă sînt de două tipuri : continue (exemplu, ritmuri de funcționare pentru puțurile de producție și de pompare, adîncimea și presiunea în puțuri etc.) și binare (numărul, locul și timpul forării de noi puțuri, conectarea/deconectarea lor la/de la rețeaua existentă etc.).

Criteriul de optimizare ales este factorul de recuperare a petrolului, iar în cazul considerării unor criterii suplimentare (de exemplu, costul exploatării) ele sînt tratate ca restricții, problema rezolvată fiind în final monocriterială.

Dimensiunea mare a modelului (sute de variabile), nelinearitatea, existența simultană de variabile continue și discrete, precum și informația insuficientă generează în final o problemă de complexitate foarte mare. Ea se poate rezolva prin aproximații succesive (de exemplu, metoda variațiilor locale), la fiecare iterație soluționîndu-se o problemă de optimizare lineară auxiliară, în general de mare complexitate deoarece are variabile mixte și este de mari dimensiuni. Fezabilitatea procedurii este în mare măsură dependentă de precizia cu care se rezolvă problemele lineare auxiliare.

În lucrarea 11.3/B2 este sugerată o metodă de rezolvare a subproblemelor lineare care folosește elemente ale teoriei sistemelor multivariabile și utilizează din plin specificitatea modelului cîmpului de petrol ca sistem multivariabil. Această metodă este analogă metodei simplex aplicată la spații funcționale, combinată cu metoda variațiilor locale. Astfel, dificultățile datorite dimensiunilor sistemului sînt depășite, iar variabilele continue sînt separate de cele discrete. În continuare variabilele discrete sînt tratate cu metode combinatoriale în contextul abordării generale a proiectării optime a rețelelor cu mai multe componente. Prin aceeași abordare, problema de optimizare lineară poate fi interpretată ca o problemă de planificare pe termen scurt.

Soluția problemei globale poate fi considerată ca proiectarea optimă a exploatării cîmpului petrolifer. Prin abordarea propusă în lucrarea 11.3/B2 conducerea tehnologiei de recuperare a petrolului este făcută compatibilă și poate fi corelată cu proiectarea construcțiilor de suprafață care urmăresc dezvoltarea sistemului de exploatare a cîmpului petrolifer.

Considerînd optimizarea completă a exploatării unui rezervor petrolifer ca un ansamblu de probleme de optimizare intercorelate, în *lucrarea 11.3/B3* se abordează problema determinării bazei teoretice de calcul al celui mai bun mod de injectare a unui fluid de recuperare sporită a petrolului dintr-un rezervor. Criteriul de optimizare este minimizarea costului substanțelor chimice injectate și maximizarea petrolului recuperat. Obiectivele criteriului se exprimă ca o funcțională cost, sau indice de performanță, care trebuie minimizată. Variabila de comandă este compoziția sau starea fizică a fluidelor injectate. Astfel, problema de optimizare urmărește determinarea politicii de injectare care duce la minimizarea funcției de cost, respectînd restricțiile egalitate diferențiale care descriu dinamica sistemului. Calculul teoretic arată că utilizînd teoria sistemelor cu parametri distribuiți este posibilă determinarea unei astfel de soluții.

În afara sistemului de control al calității într-un LBC referit anterior (11.3/A3), alte două comunicări prezentate în *secțiunea 11.3* abordează conducerea unor procese tehnologice, din industria sticlei (11.3/A2) și pentru un accelerator complex (11.3/A5). Sînt de reținut atît metodele matematice utilizate — de mare eficiență pentru obținerea răspunsului în timp real — cît mai ales organizarea sistemelor de conducere care asigură cadrul necesar pentru succesul implementării.

În industria sticlei, după procesul de tragere în foi, tăierea la dimensiunile cerute prin comenzi poate fi optimizată în scopul eliminării defectelor de mici dimensiuni. Acestea sînt constatate prin dispozitive speciale, poziția lor fiind transmisă unui calculator de proces care controlează tăierea de așa manieră încît să elimine defectele minimizînd pierderile de material și asigurînd obținerea de plăci în dimensiunile cerute.

Tăierea foilor de sticlă se face în plăci dreptunghiulare de diferite dimensiuni, conform cererii. Operația presupune tăieri „verticale", apoi „orizontale", la dimensiunile solicitate. Neținînd seama de existența și poziția defectelor, un mare număr de plăci vor fi rebutate. O economie însemnată se face tăind plăcile așa încît defectele, sesizate prin detectoarele plasate înaintea dispozitivelor de tăiere, să fie evitate ; printr-o combinare optimă a diferitelor dimensiuni de plăci, precum și printr-o dispunere optimă a acestora se poate minimiza pierderea de material

eliminat odată cu defectele. Această problemă este tratată în lucrarea 11.3/A2 printr-o strategie ingenioasă de combinare a doi algoritmi într-un sistem condus de un calculator de proces. Unul este „algoritmul de tăiere optimă", care așează optim, în spațiul unei foi la care au fost determinate defectele și poziția lor, două dimensiuni de plăci date (se demonstrează rezultate superioare față de considerarea unei singure dimensiuni a plăcilor). Așezarea se face astfel încît să se evite cu pierderi minime defectele, iar plăcile să se poată obține prin succesiuni de o tăiere verticală plus mai multe tăieri orizontale. Problema este complicată de curgerea semi-infinită a foii de sticlă.

Al doilea algoritm este un „algoritm de combinare optimă", care asigură selectarea celei mai bune perechi de plăci pentru o anumită distribuție de defecte.

Datele de intrare folosite sînt : viteza și lățimea foii de material, distribuția de probabilitate a localizării defectelor, distanța între senzorul defectelor și dispozitivul de tăiere verticală, dimensiunile și numărul de plăci cerute. Pe baza acestor date algoritmul de combinare optimă determină o pereche de sortimente de plăci care va fi produsă la un moment dat. În funcție de localizarea reală a defectelor, aceste plăci sînt așezate în foaie de către algoritmul de tăiere optimă, determinînd tăierile verticale/orizontale, care sînt transmise dispozitivelor de execuție. La epuizarea numărului solicitat de plăci dintr-o dimensiune a perechii, reintră în funcție algoritmul de combinare optimă. În afara prezentării algoritmilor, [11.3/A2] susține prin date experimentale valabilitatea și superioritatea algoritmilor propuși, față de alte metode abordabile.

Sistemul de conducere a unui accelerator complex de 28 GeV [11.3/A5] este compus dintr-o rețea de 20 minicalculatoare (calculatoare de control consolă, calculatoare frontale și pentru operații specializate, ex. gestiune mesaje) și 50 microcalculatoare cuplate la proces prin interfețe CAMAC (realizează în special funcții de control în interfețele specializate, activități de timp real, de prelucrare primară a datelor). Totul este operat de la 5 console generale. Software-ul utilizat este de două categorii : de sistem (sisteme de operare, control rețea, limbaje de programare) și de aplicație (de la facilități dialog om-calculator pînă la algoritmi de control al hardware-ului de proces).

Sistemul este ierarhizat pe 6 nivele (de tip „layer") și este realizat din module cu mare grad de independență. Cele 6 nivele sînt : module de interfață (protocoale de control standard pentru interfețele CAMAC proces-calculator) ; module de control al fiecărui tip de echipament (surse, pompe etc.) ; module de control pentru variabilele de proces complexe (exemplu, controlul fasciculului de electroni) ; module pentru funcții de comandă a procesului în condițiile date de operator ; module pentru dialogul cu operatorul ; module specifice pentru controlere și prelucrarea primară de date. Aceste module acționează sub supervizarea unor funcții de conducere locală subordonate obiectivelor globale și asigură : activarea/oprirea modulului, alocarea resurselor, prevenirea conflictelor, asigurarea sincronizării cu evenimentele specifice, afișarea mesajelor, controlul erorilor.

Sistemele de tipul celui prezentat sînt de o mare complexitate. Cum pot fi învățați inginerii să construiască astfel de sisteme ? Pînă la ce punct se poate metodologiza analiza și sinteza sistemelor complexe ? Acestea sînt întrebări la care *lucrarea 11.3/A6* încearcă să dea un răspuns. Lucrarea urmărește dezvoltarea unei metodologii de înțelegere, standardizare și îmbunătățire a procesului de proiectare a sistemelor de comandă complexe. Pleclnd de la premisa că cine poate analiza și înțelege structura de comandă a unui proces oricît de complex este capabil să dezvolte un mod bine organizat de sinteză a unui sistem de comandă asociat, autorii descompun în pași etapele de analiză și sinteză, exemplificînd pe cazul unui sistem industrial. Sistematizarea oferită este utilă în special inginerilor care fac primii pași în proiectare.

Care este însă efortul construirii unui sistem de conducere bazat pe utilizarea calculatorului ? Se poate estima eficiența așa-numitului software de aplicație ? Software-ul în general și software-ul de aplicație în particular suferă neavînd în spate experiența unei discipline inginerești îndelung încercate. Așa cum se apreciază în *lucrarea 11.3/A5*, lipsa sa de tehnici riguroase de evaluare a costului care să cîștige încrederea utilizatorului face ca orice efort onest de dezvoltare a software-ului să fie privit cu o mare doză de scepticism. De aceea, considerăm utilă și instructivă evaluarea pe care o face lucrarea 11.3/A5 pentru un sistem de conducere complex pe baza unui formalism cunoscut sub numele de Barry Boehm. Acest model a fost aplicat satisfăcător la un mare număr de proiecte de dezvoltare software, lucru dovedit și de cazul

concret considerat, în care estimarea costului făcută prin diferite alte mijloace, inclusiv intuitiv, a fost ulterior confirmată atît de formalismul referit, cît şi de costul real al proiectului. Formalismul presupune considerarea unui efort nominal bazat pe numărul estimat de instrucţiuni software precum şi pe modul de organizare în echipa de programare. Acest efort nominal este apoi ponderat cu multiplicatori de efort pentru a se obţine efortul efectiv. Aceşti multiplicatori particularizează proiectul relativ la obiectivul urmărit, hardware-ul implicat, personalul implicat şi caracteristicile de implementare.

SISTEME EXPERT

Ing. *C. Vasiliu*

I.T.C.I.

1. INTRODUCERE

Teoria sistemelor automate, în mod tradițional considerată ca parte a unei discipline analitice de sine stătătoare, a înregistrat în ultimii ani un proces de transformare calitativă, atît în ceea ce privește aparatul formal și procedural cu care se operează, cît și aria de cuprindere practică propriu-zisă. Dacă admitem că realizarea unui sistem de conducere automată presupune parcurgerea unor etape bine delimitate, cum ar fi : a) modelarea-identificarea procesului condus : b) analiza-simularea modelului ; c) proiectarea legii de comandă și d) implementarea sistemului de conducere, atunci vom constata — prin prisma observației de mai sus — că mai ales etapele a, b, c au beneficiat de acumulări importante (idei, concepte, proceduri de proiectare, algoritmi de conducere automată avansată), în timp ce impactul asupra implementării (etapa d) și operării sistemului de conducere automată a rămas mai degrabă modest (cel mai notabil lucru fiind, probabil, tranziția de la implementările analogice la cele numerice).

Să considerăm exemplul regulatorului clasic proporțional-integrator-derivativ (PID). În acest caz valoarea mărimii de comandă u (·) este furnizată de o relație de tipul :

$$u(t) = \left[e_a^*(t) + \frac{1}{T_i} \int^t e(s)\, ds + T_d \frac{de(t)}{dt} \right] \qquad (1)$$

Aceasta este o ecuație lineară care permite studiul comportării sistemului de conducere automată și ai cărei parametri se pot determina prin proceduri bine stabilite.

Dar, în implementarea regulatorului PID, relația (1), deși absolut necesară pentru calculul mărimii de comandă, se dovedește insuficientă ; multe aspecte „practice" importante nu sînt surprinse de aceasta, ca de exemplu :

— interfața cu operatorul tehnolog ;

— comutarea automat/manual ;

— regimurile tranzitorii care apar la modificarea parametrilor procesului (punctului de operare) ;

— efectele introduse de nelinearitățile elementelor de execuție ;

— saturarea termenului integral.

Asemenea probleme sînt de obicei scufundate în considerații practice de proiectare și numai foarte rar tratate și din punct de vedere teoretic. Realizarea unui regulator PID operațional presupune, desigur, implementarea formulei (1) dar și o anumită logică euristică pentru rezolvarea aspectelor menționate, aceasta constituind, de fapt, contextul logic al implementării relației (1). Un rol și mai important revine euristicii în cazul regulatoarelor multivariabile sau al regulatoarelor autoacordabile.

Această constatare se referă la activitățile asociate sistemelor de conducere automată de la nivele elementare ale unui sistem de conducere complex, ierarhizat (adică la buclele izolate). Să observăm însă că euristica joacă un rol și mai important în rezolvarea problemelor de nivel superior (adică în conducerea unui proces cu toate componentele de nivel inferior în interacțiune). La nivelul superior problema de conducere automată se poate formula în termeni de programare dinamică : stările sînt considerate în acest caz pe ansamblul procesului, iar stadiile reprezintă perioade între modificările legilor de comandă aplicate subsistemelor de nivel inferior. Deciziile corespund selectării strategiilor de conducere automată (tipul legilor de comandă, de estimare etc.) impuse asupra componentelor de nivel inferior. Criteriile de performanță de asemenea îmbracă un caracter global și pot deveni multiple.

2. LOGICA EURISTICĂ ÎN CONDUCEREA AUTOMATĂ

Aşa cum am menţionat, recursul la logica euristică se impune cu necesitate pentru proiectantul de sisteme automate, fie că este vorba de regulatoarele PID obişnuite, de sistemele de reglare hibride (conţinînd circuite logice) sau de cele avansate.

În implementare, partea aferentă euristică se transpune prin blocuri de selecţie („if-then-else": „dacă-atunci-altfel") sau de acţiuni dependente de situaţie („case" : „în cazul că"). Ca volum, codul (programul-sursă) respectiv poate depăşi codul formulei (1) ; în plus, testarea, depanarea şi eventual modificarea părţii din programul de conducere automată asociate „euristicii" se poate dovedi relativ complicată.

Din punctul de vedere al ingineriei programelor de conducere automată a proceselor apare deci justificată căutarea unor noi instrumente de implementare.

Înainte de a introduce un astfel de instrument vom mai prezenta cîteva consideraţii decurgînd dintr-o inversare de perspectivă : anume, dacă acceptăm că algoritmii de conducere automată includ logică euristică, să vedem ce ar putea aduce euristica în dezvoltarea sistemelor de conducere automată industriale.

Vom apela din nou la un exemplu : sistemele adaptive cu model de referinţă sau cele cu autoacordare. Algoritmii de comandă respectivi pot fi interpretaţi ca algoritmi de gradient local : pornind de la estimări iniţiale suficient de bune ale ordinului sistemului, perioadei de eşantionare şi ale parametrilor modelului, algoritmul poate ajusta parametrii regulatorului astfel încît să rezulte un sistem în circuit închis satisfăcător, cu condiţia ca estimările iniţiale să fie suficient de apropiate de valorile reale. Algoritmii menţionaţi posedă chiar şi proprietăţile de urmărire, cu condiţia ca parametrii să se modifice lent în timp. Algoritmii nu funcţionează dacă estimările iniţiale sînt depărtate de valorile reale (cazul cel mai tipic este perioada de eşantionare). De aici necesitatea unor algoritmi cu domeniu de operabilitate mai mare (mai siguri, într-un anumit sens), chiar în detrimentul proprietăţilor locale.

Din acest punct de vedere pare atractivă exploatarea posibilităţii de a utiliza — într-o structură evoluată de conducere — algoritmi cu proprietăţi diferite. Acest lucru reclamă facilităţi de combinare a diferiţilor algoritmi de conducere automată, astfel încît să se satisfacă criterii de performanţă adecvate fiecărui regim de funcţionare.

Formulată în alţi termeni — care se vor preciza în continuare — această problemă îşi găseşte o rezolvare, cel puţin potenţială, în aplicaţiile sistemelor expert.

3. SISTEME EXPERT

Sistemele expert reprezintă un domeniu în plină dezvoltare al inteligenţei artificiale ; obiectivul acestora este de a modela cunoştinţele şi procedurile utilizate de experţii umani în rezolvarea problemelor dintr-un domeniu bine stabilit. Pentru familiarizarea specialistului în automatică vom prezenta în continuare o scurtă introducere în sistemele expert. Ideile expuse sînt preluate după (I. Georgescu, 1983), unul din exponenţii recunoscuţi ai şcolii româneşti de inteligenţă artificială.

Capabilităţile funcţionale ale sistemelor expert pot fi clasificate după cum urmează :

— *funcţii cognitive*, care oferă sistemului capacitatea de a trata cunoaşterea de tip expert ;

— *funcţii rezolutive*, care se utilizează în procesul de rezolvare a problemelor pe baza cunoştinţelor din domeniul de expertiză şi a informaţiilor furnizate de utilizator asupra contextului problemei considerate ;

— *funcţii explicative*, cu rolul de a explicita judecăţile şi inferenţele care au condus la soluţiile preconizate ;

— *funcţii de comunicare*, care permit sistemului expert să comunice cu utilizatorul uman sau cu alte sisteme expert.

Potenţialul de rezolvare a problemelor de către sistemele expert depinde de capacitatea de a memora, actualiza şi manipula volume considerabile de cunoştinţe specifice domeniului, de flexibilitatea asigurată în reprezentarea cunoştinţelor indiferent de complexitatea acestora, de posibilitatea de a apela la strategii de planificare şi rezolvare evoluate ca şi de capacitatea de a-şi îmbunătăţi performanţele funcţionale prin învăţare din experienţă.

Indiferent de cerinţele particulare ale domeniului real de expertiză, un sistem expert se poate considera ca fiind alcătuit din următoarele elemente :

— *sistemul cognitiv*, destinat să memoreze cunoştinţele într-un sistem de memorare special organizat denumit *baza de cunoştinţe*, să caute cunoştinţe parţiale direct specificate, prin identificarea de simboluri sau referite indirect, prin proprietăţile asociate şi valorile atribuite sau deduse din alte cunoştinţe şi de asemenea să întreţină baza de cunoştinţe conform evoluţiei domeniului de expertiză ;

— *sistemul rezolutiv*, destinat să rezolve problemele specificate de către utilizator prin aplicarea unor proceduri şi operatori asupra obiectelor reale extrase din formularea problemei şi din baza de cunoştinţe, într-o secvenţă adecvată şi pe baza unei strategii de control care furnizează finalmente soluţia problemei ;

— *sistemul de comunicaţie*, care constituie interfaţa dintre utilizator şi sistemul expert, prin intermediul unor procesoare adecvate de limbaj şi al unor proceduri de intrare-ieşire.

Suportul obiectual al unui sistem expert, organizat în baza de cunoştinţe, include cunoştinţe *declarate*, care se autoexplicitează şi cunoştinţe *generative*, care sînt în esenţă mecanisme productive pentru obiecte, acţiuni, relaţii, stări şi evenimente, capabile să genereze o entitate cu parametri specificaţi sau un set finit de entităţi cu proprietăţi specificate.

Suportul procesual al unui sistem expert este organizat într-o ierarhie de patru nivele :

— nivelul mecanismelor de bază, care include mecanisme inferenţiale pentru aspectele logice şi mecanisme operaţionale pentru aspectele algebrice ;

— nivelul procesoarelor de acţiune, care interpretează cunoştinţele ce descriu acţiuni ;

— nivelul procesoarelor de control, concepute ca organe de decizie asupra strategiei de control pe care sistemul expert trebuie să o urmeze în rezolvarea unei probleme ;

— nivelul procesoarelor de metacontrol, destinate să controleze, adapteze şi actualizeze funcţionarea procesoarelor de la nivelele subordonate.

Prin control în acest context se înţelege un procedeu de supervizare a proceselor de tip acţiune.

În ceea ce priveşte reprezentarea cunoştinţelor, cea mai importantă opţiune conceptuală în proiectarea unui sistem expert, există mai multe metode, fiecare dintre acestea avînd limite şi influenţe asupra extinderii domeniului de expertiză, asupra complexităţii cunoştinţelor, asupra flexibilităţii în manipularea cunoştinţelor şi finalmente asupra performanţelor sistemului expert ca atare. Dintre metodele de reprezentare a cunoştinţelor vom menţiona :

— limbaje bazate pe calculul predicatelor de ordinul întîi (formalism direct interpretabil de către componentele rezolutive ale sistemului expert) ;

— reprezentarea procedurală (în plus faţă de caracteristicile şi proprietăţile structurale se includ şi descrieri ale modalităţilor de utilizare a cunoştinţelor în acţiuni şi procese) ;

— reprezentarea prin reţele semantice (prin excelenţă un instrument de reprezentare de tip relaţional, în care nodurile se asociază obiectelor, acţiunilor, stărilor sau altor entităţi abstracte din domeniul de expertiză, iar arcele descriu relaţiile dintre acestea) ;

— reprezentarea prin sisteme de producţie (conceptual, un sistem de producţie se compune din : a) un set finit de reguli de producţie de tipul "DACA-condiţie-ATUNCI-acţiune" ; b) un set finit de date specificînd contextul de aplicare a regulilor şi c) un interpretor care decide iniţierea, secvenţa de reguli şi oprirea executării producţiilor) ;

— reprezentarea prin obiecte structurate („cadre" reprezentînd descrierea unui obiect, acţiuni sau situaţii stereotipe pe care o anticipăm atunci cînd considerăm anumite cunoştinţe).

Să examinăm acum domeniul de expertiză al inginerului automatist. Baza de date va conţine informaţii faptice, evidenţe (în sensul unor înregistrări ale evoluţiei), ipoteze şi obiective. Ca faptice vom considera, de exemplu, datele statice : toleranţele traductoarelor, limitele de operare, limitele de alarmă pentru parametrii procesului, restricţiile de înlănţuire pentru acţiunile de comandă, configuraţia de utilaje care concură la realizarea procesului tehnologic respectiv. Ca evidenţe se pot considera datele dinamice de la senzori, rapoartele şi testele de laborator ş.a. Faţă de acestea vom observa următoarele particularităţi : diversitatea în tip, afectarea de zgomot, întîrzierea (evidenţierii în raport cu momentul producerii), incompletitudinea şi chiar caracterul contradictoriu.

Desigur, inginerul automatist are capacitatea de a elabora soluţii (ipoteze) pentru rezolvarea acestor complicaţii. În sistemele expert ipotezele se vor genera şi memora în baza de date ; aşa cum s-a menţionat, ipotezele sînt astfel prezentate încît utilizatorul poate verifica raţionamentul sistemului.

Obiectivele reţinute în baza de cunoştinţe pot fi atît statice, cît şi dinamice în natură. Statice sînt, de exemplu, obiective de genul "optim staţionar" sau "sistem stabil", în timp ce obiectivele dinamice sînt stabilite on-line prin comenzi externe de către sistemul de conducere automată.

Tipurile de cunoştinţe necesare sînt :
— caracterizarea algoritmilor de comandă şi estimare disponibili ;
— indicatoare pentru invocarea funcţiilor de supervizare, planificare sau diagnosticare ;
— instrucţiuni propriu-zise pentru supervizare, planificare sau diagnoză.

Pentru aplicaţiile de conducere automată, regulile de producţie şi obiectele structurate par a fi cele mai potrivite metode de reprezentare a cunoştinţelor. De exemplu, aşa cum am menţionat deja, o regulă de producţie se descrie în general prin „dacă-situaţie-atunci-acţiune".

"Situaţie" reprezintă un set de elemente ale bazei de cunoştinţe, iar "acţiune" poate însemna activarea unui regulator, specificarea unui nou obiectiv (criteriu de performanţă) etc. Obiectele structurate reprezintă structuri de date asignate situaţiilor vizuale ; informaţiile incluse se referă la instrucţiunile de utilizare (a obiectului structurat), ce se va întîmpla la momentul următor, ce trebuie făcut la momentul următor, precondiţii necesare pentru o anumită operaţie şi altele.

În funcţionarea reală a sistemului de conducere automată un defect oarecare, o schimbare a parametrilor calitativi ai procesului de producţie sau orice altă modificare necesită o secvenţă de acţiuni pentru a alinia procesul la noile cerinţe. Fiecare pas al planului de modificare a evoluţiei procesului presupune o acţiune de ajustare, iar acţiunile stabilite la un anumit pas trebuie să nu interfereze cu precondiţiile pentru acţiunile următoare.

Dacă sînt multe acţiuni şi multe secvenţe posibile, elaborarea planului de conducere automată presupune căutarea drumului care conduce la obiectivul fixat printr-o reţea relativ mare.

Planificarea acţiunilor în condiţiile unui sistem complex cuprinde o serie de elemente specifice : incertitudine în starea şi modelul procesului, în efectul acţiunilor de comandă aplicate.

Deşi programarea dinamică statistică poate trata, în principiu, această problemă, totuşi această abordare algoritmică de tip căutare globală se poate dovedi nerealizabilă din punct de vedere practic.

4. REGULATORUL ADAPTIV EXPERT

Pentru a ilustra conceptele introduse vom prezenta succint, ca exemplu, conducerea în regim staţionar a unui proces industrial.

Fie, pentru simplitate, cazul unei singure bucle de reglare cu obiectivul de a menţine fluctuaţiile variabilei de ieşire, apropiate de o mărime de referinţă dată. Dacă dinamica procesului şi perturbaţiile ar fi cunoscute, atunci un regulator de minimă varianţă (dispersie) ar constitui soluţia de conducere automată. Într-adevăr dacă procesul este dat de modelul eşantionat :

$$Ay(t) = Bu(t-d) + Ce(t) \tag{1}$$

unde u (·) este mărimea de comandă, y (·)— mărimea de ieşire, e (·)— un zgomot alb iar A, B, C sînt polinoame în operatorul de anticipare, atunci legea de comandă optimală este reprezentată de :

$$Ru = -Sy \tag{2}$$

unde polinoamele R, S sînt date de :

$$z^{d-1}CB = AR + BS \tag{3}$$

Operaţiile necesare pentru a obţine şi menţine legea de comandă de minimă varianţă (dispersie), dată de relaţia (2), într-un mod sigur vor fi prezentate în lumina consideraţiilor anterioare.

1) În terminologia sugerată, acţiunea "comandă de minimă varianţă" constituie obiectivul principal al sistemului. Vom avea ca precondiţii şi cerinţe : modelul procesului (1), modelul perturbaţiilor, intervalul de eşantionare, condiţia ca procesul să fie de fază minimă (toate zerourile să fie situate în interiorul cercului unitate).

2) Un aspect particular al legii de comandă de varianță minimă îl constituie anularea zerourilor procesului, ceea ce poate conduce la un regim oscilant dacă zerourile nu sînt suficient de amortizate. Apare astfel ca necesară includerea unui detector de oscilații în schema practică.

3) O manieră vizuală de a stabili dacă procesul tehnologic evoluează sub o comandă de minimă varianță (dispersie), este de a verifica dacă ieșirea este ca însăși un proces de medie alunecătoare :

$$y(t)=\lambda[e(t)+\ldots+f_{d-1}e(t-dh+h)]$$

unde F=R/B iar h este intervalul de eșantionare. De aici necesitatea unui *supervizor de minimă varianță*, care poate funcționa pe baza calculului funcției de corelație a ieșirii.

Să analizăm ce se întîmplă în cazul că unele condiții nu sînt îndeplinite.

a) Dacă modelul procesului nu este disponibil, atunci se poate folosi un regulator autoacordabil, care este în esență un estimator (și care converge, în anumite condiții, către regulatorul de minimă varianță). Prin urmare, *un estimator de parametri* va fi inclus ca operator. Acesta va funcționa corespunzător numai dacă datele experimentale sînt obținute atunci cînd procesul este bine excitat : este deci, nevoie de un *supervizor de excitare* (care va determina energia semnalului de intrare în domeniul util de frecvențe).

b) În cazul că excitarea procesului nu este suficientă, sînt posibile două opțiuni :

b1) oprirea estimării și

b2) introducerea unor semnale perturbatoare (dacă procesul tehnologic permite acest lucru).

Aceasta implică existența unui *generator de semnale de perturbație*, pentru care sînt necesare cel puțin specificarea domeniului de frecvențe și a nivelului admis al perturbațiilor. Aceste elemente se pot deduce din cunoașterea orizontului de predicție dh).

c) Anumite date a priori sînt necesare și pentru regulatorul autoacordabil :

h — perioada de eșantionare ;
d — întîrzierea (în multipli de h) ;
nr — gradul polinomului R ;
ns — gradul polinomului S ;
λ — factorul de uitare ;
θ_0 — estimația inițială ;
p_0 — covarianță inițială ;
uh, ul — limitele superioară și respectiv inferioară ale mărimii de comandă.

Acestea, la rîndul lor, antrenează și ele anumite condiții :

— d și h : sistemul în buclă închisă poate deveni instabil dacă acești parametri sînt subestimați. Rezultă așadar că este necesar un *supervizor de stabilitate* ;

— ns și nr : valorile gradelor trebuie să fie suficient de mari (validitatea lor se poate preciza prin calculul funcțiilor de covarianță $\gamma_{yy}(\tau)$ și $\gamma_{yu}(\tau)$. De aici necesitatea unui *supervizor de grad*.

— produsul dh : o estimare robustă a acestuia se poate obține pe baza amplificării critice K_c și a perioadei critice t_c, conform relației $d\cdot h=t_c/2$.

Vom adopta în acest caz un *estimator de tip* K_c—t_c (care furnizează date utile și pentru estimarea altor parametri).

d) Se pot lua în considerare și alte funcțiuni sau facilități. De exemplu, dacă dinamica procesului depinde de anumiți parametri (cum ar fi ritmul de producție), atunci se poate avea în vedere programarea amplificării și se pot deci introduce funcțiuni ca : "netezește și memorează parametrii regulatorului", "actualizează parametrii regulatorului" și "testează ipotezele de programare" (a amplificării). Un *supervizor instruibil* va asigura desfășurarea corespunzătoare a tuturor acestor funcțiuni.

Rezumînd cele expuse, implementarea unui regulator adaptiv expert presupune următoarele grupe de operatori :

Program de conducere general
    Supervizor-stabilitate
    Calculul-valori -medii-și-variante (dispersii)
Monitor conducere automată
Sistem de rezervă
    Regulator PID
    Estimator-Kc-tc

Regulator de minimă varianţă cu amplicare constantă
Reglare-de-minimă-varianţă
Supervizor-de-minimă-varianţă
Detector-de-oscilaţii
Supervizor-de-grad
Estimare :
Estimator-de-parametri
Supervizor-de-estimare
Supervizor-de-excitare
Generator-semnal-perturbaţii
Detector-de-salt
Regulator autoacordabil
Reglare-autoacordabilă
Instruire
Actualizare-parametri-regulator
Netezire-şi-memorare-parametri-regulator
Testarea-ipotezelor-de-programare.

5. REALIZĂRI ŞI PERSPECTIVE

Pe baza lucrărilor celui de al 9-lea congres IFAC de la Budapesta, 1984, precum şi a unor cercetări proprii (Vasiliu C. şi Dumitru M., 1983 ; Ceangă E. şi Vasiliu C., 1984) am încercat să prezentăm în paragrafele anterioare o soluţie de perspectivă la o problemă tot mai stringentă a automaticii industriale : anume, realizarea de sisteme de automatizare flexibile, cu elemente de inteligenţă artificială.

Cu excepţia consideraţiilor privitoare la sistemele expert, expunerea a urmărit îndeaproape *lucrarea 11.4/B2.*

Vom trece în revistă alte probleme conexe abordate în lucrările congresului amintit (*secţiunile 11.4/A* şi *11.4/B*).

Aspecte legate de rezolvarea problemelor de modelare şi comandă în sistemele om-maşină sînt examinate în *lucrarea 11.4/A1.* Natura interacţiunilor între comandă, planificare şi detecţia şi anihilarea erorilor sînt exemplificate, pentru cazul unui autovehicul şi al unei aeronave. Autorii susţin ideea unui sistem expert ca fiind cel mai adaptat cerinţelor umane în situaţiile examinate.

Conducerea în regim supervizor şi dirijarea situaţiilor anormale (fault management) fac de asemenea obiectul *lucrării 11.4/A3.* Se arată că un model al operatorului uman acţionînd în regim supervizor depinde de modelul sistemului supervizat, de perturbaţiile acestuia şi de obiectivul procesului de supervizare ; un model analog în cazul unor situaţii anormale nu se poate construi. Soluţia propusă de autori pentru completarea acestei lacune constă într-o abordare de tip normativ bazată pe teoria utilităţii.

Un aspect esenţial care condiţionează succesul în proiectarea sistemelor om-maşină (cum este şi cazul unui operator uman faţă de un sistem de conducere automată) îl constituie înţelegerea modului în care omul formulează ipoteze şi concluzii, identifică posibile variante de acţiune şi analizează impactul acestor variante, precum şi integrează consecinţele acţiunilor în concordanţă cu un sistem de evaluare. *Lucrarea 11.4/A4* conţine o privire de ansamblu asupra eforturilor recente în domeniul reprezentării cunoştinţelor, o atenţie particulară fiind acordată sistemelor de producţie.

Pentru a obţine modele cît mai realiste ale operatorului uman aflat în postura de a prelucra informaţii şi a lua decizii este necesar să se modeleze şi memoria acestuia, fie că este internă procesului de decizie, fie că este externă acestuia (în forma unei baze de date, de exemplu), Un model de tip informaţional al memoriei permanente este prezentat, împreună cu descrierea accesului la informaţiile memorate, pentru un decident uman în *lucrarea 11.4/A5.*

Sistemele de decizie asistată de calculator sînt abordate în *lucrarea 11.4/A6,* în contextul unor concepte de psihologie, cibernetică şi sisteme automate.

Pe aceeaşi linie de evoluţie, adică în completarea abordărilor clasice, în lucrarea 11.4/B1 se utilizează metodele teoriei mulţimilor vagi pentru descrierea strategiei unui operator uman.

Exemplele prezentate sînt simple, dar consacră abordarea "fuzzy" ca o soluţie în cazul în care metodele clasice nu pot funcţiona.

În sfîrşit, *lucrarea 11.4/B4* încearcă trasarea unui cadru conceptual pentru proiectarea sistemelor de automatizare complexe. Pornind de la necesitatea ca datele abstracte şi adesea informale, generate pe parcursul etapelor preliminare de proiectare a unui sistem să fie astfel documentate încît să poată fi utilizate în etapele ulterioare (nu numai de proiectare, ci chiar de operare a sistemului respectiv) autorii propun ca instrument un limbaj dedicat, de nivel înalt, denumit SML (System Modeling Language), implementat pe un calculator VAX 11/750 cu sistemul de operare VMS.

BIBLIOGRAFIE

1. *Georgescu I.* (1983), **An Introduction to Expert Systems**, Report AI-FDO30, ICI.
2. *Vasiliu C., Dumitru M.* (1983), **Tehnici de recunoaştere a formelor şi de inteligenţă artificială în proiectarea sistemelor**, Simpozionul de Modelare şi Optimizarea Sistemelor, Universitatea Galaţi.
3. *Ceangă E., şi Vasiliu C.* (1984), **Learnind Systems Using Control Situations Recognition**, Raport intern, ICI.

CONDUCEREA DE CĂTRE OPERATOR UMAN A VEHICULELOR ŞI TELEMANIPULATOARELOR

Ing. *A. Alexandru*, Ing. *L. Iacob*.
Ing. *P. Rădulescu-Banu*

I.T.C.I.

Pentru a înţelege comportarea umană în sisteme conduse manual este din ce în ce mai importantă investigarea pornită din interior în locul descrierii exterioare. *Lucrarea 11.4/C4* este un pas într-o astfel de investigaţie, studiind ieşirea reglată manual a unui sistem comandat manual. Se consideră că ieşirea constă din 3 componente. Prima este componenta de intrare, care se presupune a fi o funcţie continuă de timp, generată de modelul intrării în mintea operatorului. Cea de a doua este o componentă sistem, care depinde de starea curentă a erorii. Cea de a treia este zgomot. Modelul conceput pentru operatorul uman este prezentat în fig. 1.

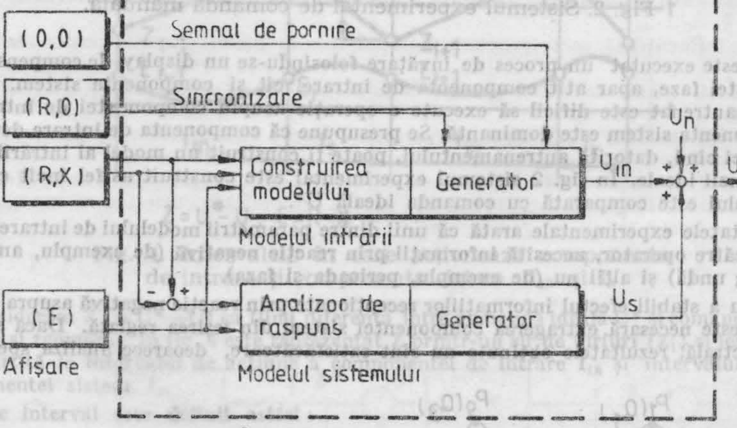


Fig. 1. Modelul conceput pentru operatorul uman.

Modelul intern este constituit mental de către operator prin identificarea caracteristicilor intrării şi dinamicii echipamentului. El generează comanda ideală şi este denumit modelul intrării, ieşirea lui fiind componenta de intrare U_{in}. Experimental, ţinînd cont de evoluţia intrării de referinţă R şi a variabilei reglate x, s-a arătat că modelul intrării are caracteristici de buclă deschisă de reglare.

Pe de altă parte, modelul sistemului este un model intern construit prin interacţiunea dintre operator şi sistem în buclă închisă de reglare, ieşirea acestui model este componenta sistem U_s, care are rolul de a preveni creşterea erorii sau a stabiliza sistemul şi de a compensa eroarea făcînd-o să fie zero.

Se presupune că strategia de comandă a operatorului uman, însoţită de operaţia de recunoaştere, este reflectată de către componenta de zgomot U_n, datorită deci unor operaţii fizio-

logice și psihologice. Se consideră că această componentă există atît în cazul reacției negative, cît și în caz contrar.

Sistemul experimental de comandă manuală este prezentat în fig. 2.

Pentru experimentări, intrarea de referință este considerată sinusoidală R=a sin ωt. Pentru a simula dinamica echipamentului se folosește un factor de amplificare proporțional G (s), frecvența ω este 0,2 ; 0,5 ; 1,0 ; 2,0 [rad/s] și amplitudinea R este calibrată la 5 cm pe un dispozitiv de afișare grafică localizat la 50 cm de operator. Controlerul este un potențiometru rotativ (de la —60° la 60°). Fiecare experiment durează 60 de secunde.

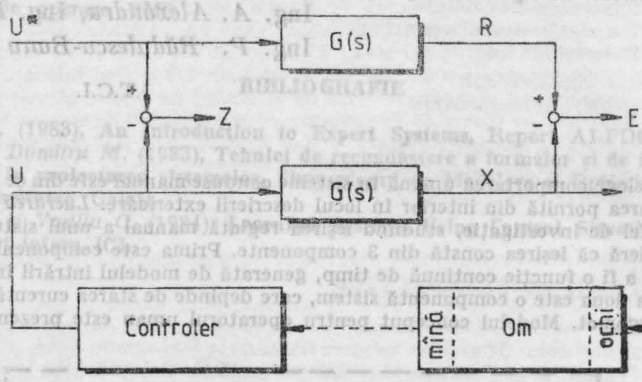


Fig. 2. Sistemul experimental de comandă manuală.

Întîi este executat un proces de învățare folosindu-se un display de compensare (E). În timpul acestei faze, apar atît componenta de intrare, cît și componenta sistem. Pentru un operator neantrenat este dificil să execute o operație asupra componentei de intrare, astfel încît componenta sistem este dominantă. Se presupune că componenta de intrare devine dominantă atunci cînd, datorită antrenamentului, poate fi construit un model al intrării corespunzător comenzii ideale. În fig. 2 sistemul experimental este construit astfel încît comanda U a operatorului este comparată cu comanda ideală U*.

Rezultatele experimentale arată că unii dintre parametrii modelului de intrare, constituit mental de către operator, necesită informații prin reacție negativă (de exemplu, amplitudinea și forma de undă) și alții nu (de exemplu, perioada și faza).

Pentru a stabili efectul informațiilor recepționate prin reacție negativă asupra componentei sistem, este necesară extragerea componentei sistem din ieșirea reglată. Dacă se folosește analiza spectrală, rezultatele obținute nu sînt satisfăcătoare, deoarece analiza spectrală este

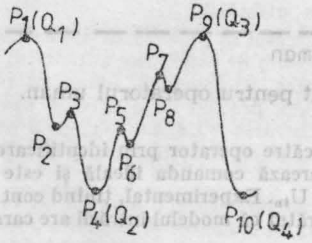


Fig. 3. Un exemplu de formă de undă constînd din 2 componente {Pi}, {Qj}.

validă pentru unde staționare, iar ieșirea reglată nu este astfel. De aceea, se acordă atenție caracteristicilor de timp ale ieșirii reglate, reprezentîndu-se eroarea E (=R—x) ca un șir de vîrfuri locale $\{P_i\}$, ca în fig. 3 și se definește perioada de timp T ca diferența $T=T_{i+1}-T_{i-1}$ între P_{i+1} și P_{i-1}.

Folosirea unui filtru pentru separarea componentelor de frecvență înaltă și joasă creează posibilitatea analizei în timp a fiecăreia din ele. Se arată că $\{Q_j\}$ — vîrfuri lente și $\{P_i\}$ sînt

legate de componenta sistem, respectiv de zgomot. Deoarece componenta sistem operează cu o frecvenţă mai ridicată decît intrarea de referinţă, ea este adesea privită drept zgomot care înrăutăţeşte comportarea sistemului. Aplicînd analiza în timp, vîrfurile $\{P_i\}$ separate de un filtru, se constată existenţa lor de la 0,3 la 0,4 secunde. Astfel, componenta care are vîrful în regiunea de la 0,3 la 0,4 secunde este superimpusă ieşirii reglate, independent de existenţa reacţiei negative sau intrării de referinţă. De aceea, această componentă este considerată drept zgomot cauzat de operaţiile fiziologice şi psihologice.

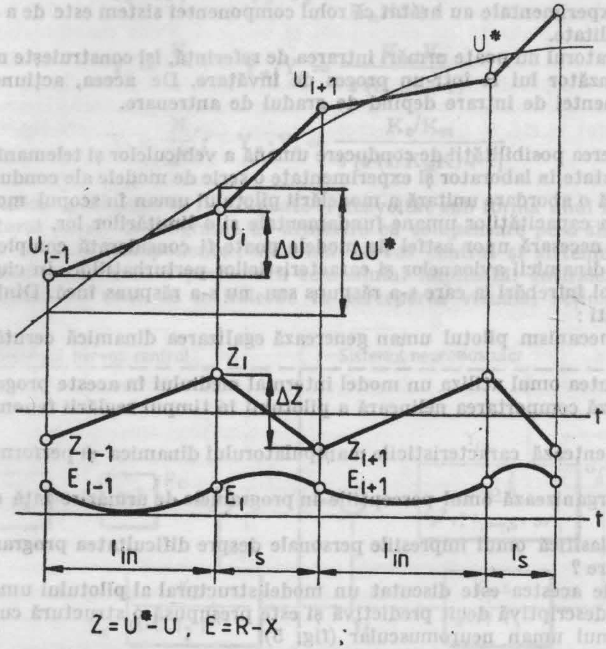


Fig. 4. Intervalele în care acţionează componenta de intrare şi componenta sistem : I_{in} şi I_s.

Considerăm Z (=U*—U) ca fiind diferenţa între comanda ideală U* şi comanda U, în care se reflectă şi zgomotul. În fig. 4 este reprezentat Z printr-un şir de vîrfuri $\{Z_i\}$ şi ieşirea reglată este împărţită în intervalul de acţiune a componentei de intrare I_{in} şi intervalul de acţiune a componentei sistem I_s.

Fiecare interval este definit astfel :

$$\dot{I}_{in}\,(\text{tip } 1):\ (E_{i+1}+E_i)(Z_{i+1}-Z_i)\geq 0 \qquad (1)$$
$$\Delta U^*\cdot\Delta U\geqslant 0$$
$$\dot{I}_s\,(\text{tip } 2):\ (E_{i+1}+E_i)(Z_{i+1}-Z_i)<0 \qquad (2)$$
$$\Delta U^*\cdot\Delta U\geqslant 0$$
$$I_s\,(\text{tip } 3):\ \Delta U^*\cdot\Delta U<0 \qquad (3)$$

unde ΔU* şi ΔU sînt variaţiile lui U* şi U în fiecare interval.

Exactitatea componentei de intrare este evaluată prin raportul ΔU/ΔU* în I_{in}. Pentru a descrie caracteristicile distribuţiei lui ΔU în raport cu ΔU* în coordonate ortogonale este definită următoarea valoare medie pentru precizia trasării

$$h_{in}=1-\sqrt{\sum_i (\Delta U_i^*-\Delta U_i)^2/\sum_i \Delta U_i^{*2}} \qquad (4)$$

În cazul în care acţionează numai componenta de intrare, Z ar trebui să tindă spre zero. Dar Z este corectat de fiecare dată, fapt care conduce la o mică variaţie. De aceea se defineşte precizia trasării îmbunătăţită de componenta sistem ca

$$h = 1 - \sqrt{\int_0^T Z^2\,dt \Big/ \int_0^T U^{*2}\,dt} \quad (5)$$

Rezultatele experimentale au arătat că rolul componentei sistem este de a corecta eroarea, conducînd la stabilitate.

Deoarece operatorul nu poate urmări intrarea de referinţă, îşi construieşte mental un model de intrare corespunzător lui R într-un proces de învăţare. De aceea, acţiunea componentei sistem şi a componentei de intrare depind de gradul de antrenare.

Pentru studierea posibilităţii de conducere umană a vehiculelor şi telemanipulatoarelor au fost dezvoltate, testate în laborator şi experimentate o serie de modele ale conducătorului uman. *Lucrarea 11.4/C5* dă o abordare unitară a modelării pilotului uman în scopul modificării modului de considerare a capacităţilor umane fundamentale şi a limitărilor lor.

Baza de date necesară unor astfel de modele poate fi considerată completă din punct de vedere al efectelor dinamicii avioanelor şi caracteristicilor perturbaţiilor. În ciuda acestui fapt au apărut recent noi întrebări la care s-a răspuns sau nu s-a răspuns încă. Dintre aceste întrebări se pot aminti:

(1) Prin ce mecanism pilotul uman generează egalizarea dinamică cerută în programele de urmărire?

(2) Cum ar putea omul utiliza un model intern al mediului în aceste programe?

(3) Ce cauzează comportarea nelineară a pilotului în timpul reglării fenomenelor cu dinamică ridicată?

(4) Cum influenţează caracteristicile manipulatorului dinamica şi performanţele pilotului uman?

(5) Cum îşi organizează omul percepţiile în programele de urmărire faţă de cele de compensare?

(6) Cum îşi clasifică omul impresiile personale despre dificultatea programului sau abilităţile în manipulare?

Ţinînd cont de acestea este discutat un model structural al pilotului uman. Modelul are mai mult o natură descriptivă decît predictivă şi este presupusă o structură cu reacţie inversă implicată de sistemul uman neuromuscular (fig. 5).

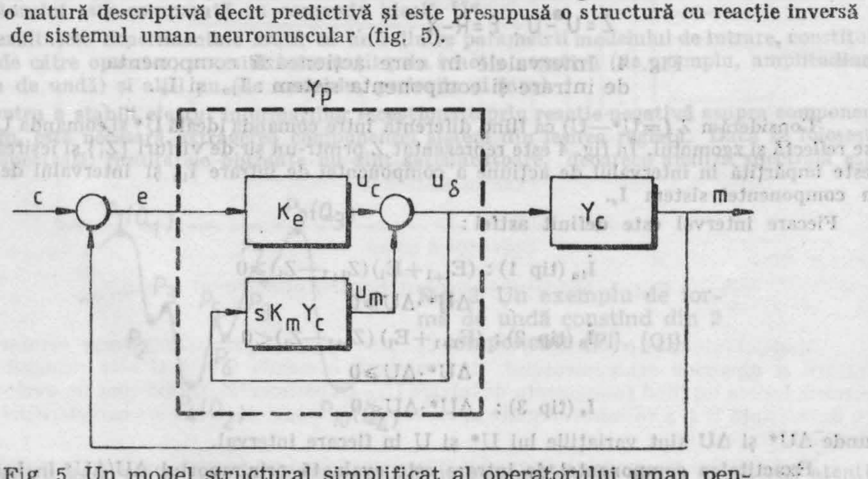


Fig. 5. Un model structural simplificat al operatorului uman pentru programe de compensare.

Modelul se bazează pe principii simple de reacţie negativă. Omul realizează egalizarea cerută de orice program printr-o reacţie negativă proprioceptivă şi nu prin vreo operaţie serială directă asupra stimulului vizual e (t). Caracteristicile generale ale acestei egalizări depind de

prelucrarea ulterioară a ieşirii manipulatorului U_δ (t) : diferenţiere, atenuare simplă sau integrare în frecvenţă în jurul frecvenţei de ieşire. În bucla interioară pentru reacţia negativă este utilizată ieşirea manipulatorului, care alimentează un element de compensare în care apare în mod explicit elementul comandat Y_c. Este interesant de utilizat modelul pentru a forma funcţia de transfer a buclei deschise pentru trei elemente comandate stereotipe : K, K/s şi K/s^2.

$$Y_c = K\ ;\quad Y_p \cdot Y_c = \frac{K_e \cdot K}{K_m K S + 1}$$

$$Y_c = \frac{K}{s}\ ;\quad Y_p \cdot Y_c = \frac{K_p \cdot K}{s\,(K_m K + 1)} \tag{6}$$

$$Y_c = \frac{K}{s^2}\ ;\quad Y_p \cdot Y_c = \frac{K_e/K_m}{s\,[s\,(1/K_m K) + 1]}$$

Modelul simplificat prezentat în fig. 5 poate fi dezvoltat sub forma unui model (fig. 6) care simulează operatorul uman în programe de urmărire cu compensare pe o singură axă. Figura 6 este împărţită în două componente : sistemul nervos central şi sistemul neuromuscular, divizare care accentuează natura prelucrării semnalului implicată. Întîrzierea de τ_0 secunde a fost inclusă ţinîndu-se cont de întîrzierile în perceperea vizuală, acţionarea motorie etc.

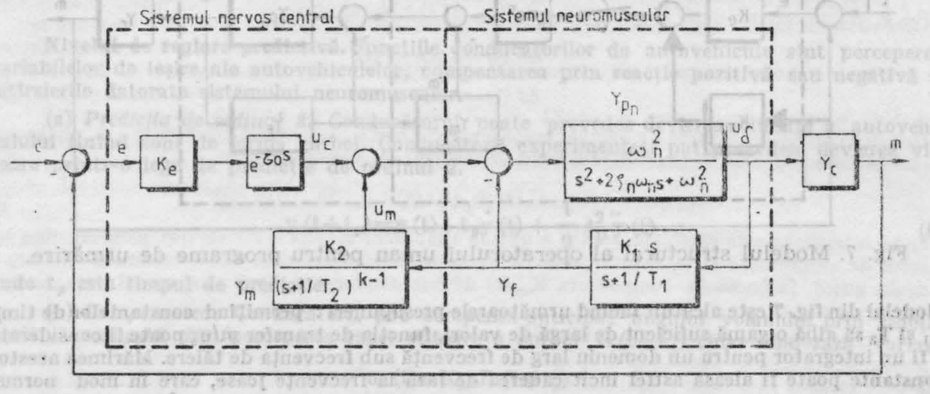


Fig. 6. Modelul structural al operatorului uman pentru programe de compensare.

Semnalul rezultant, u_c (t), asigură comanda sistemului în buclă închisă, care conţine modelul dinamicii neuromusculare în buclă deschisă a membrului care acţionează asupra manipulatorului, Y_{pn}. Celelalte componente ale sistemului includ elementele Y_f şi Y_m, care acoperă, cel puţin aproximativ, efectele combinate ale sistemului muscular, tendonului Golgi şi dinamicile asociate prelucrării de nivel înalt a semnalului. Forma lui Y_m este determinată de natura dinamicii elementului comandat Y_c în regiunea frecvenţei de tăiere în buclă deschisă. Deşi funcţia de transfer u_m/u_δ pentru fig. 6 nu este identică cu cea din fig. 5 pentru toate frecvenţele, cele două sînt echivalente în regiunea de tăiere pentru valori ale parametrilor care conduc la comportări asemănătoare ale modelului şi experimentului.

Din ceea ce se ştie despre structura sistemului uman neuromuscular, se poate concluziona că este foarte probabil ca numai necesităţile procesului de integrare să implice „activitate de calcul" la nivelele superioare ale sistemului nervos central. Cînd sînt implicate elemente de comandă de ordin superior, cum ar fi K/s^2, operatorul uman produce adesea o ieşire discretă sau pulsativă a manipulatorului. Intrările gaussiene produc adesea ieşiri ale manipulatorului cu distribuţii bimodale ale amplitudinii. Distribuţia bimodală a amplitudinii este creată adesea de comenzi de ieşire denumite comenzi „bang-bang", adică asemănătoare ieşirii unui releu. Comportarea pulsativă sau impulsivă a operatorului uman în acţionarea unor elemente de

comandă de ordin superior poate reprezenta o încercare de generare a unei ieșiri de comandă $u_c(t)$ mult mai ușor integrabilă decît cea existentă în absența pulsării, adică aplicabilă unor forme de undă a căror integrare necesită un minimum de activitate de nivel înalt a sistemului nervos central.

Scopul principal al introducerii modelului structural al operatorului uman este, după cum s-a menționat, de a modifica modul de considerare a capacităților umane fundamentale și a limitărilor lor în programele clasice de urmărire. Importanța acestui fapt a fost demonstrată de către rezultatele testelor experimentale de zbor desfășurate la NASA, care au arătat dependența dintre egalizarea realizată de către pilot și caracteristicile manipulatorului.

Dacă pe dispozitivul de afișare grafică sînt reprezentate atît eroarea cît și informațiile de intrare, atunci este denumit dispozitiv de afișare grafică de urmărire. În fig. 7 este prezentat modelul structural al operatorului uman pentru programe de urmărire (Hess, 1981), alcătuit similar celui prezentat în fig. 6. Liniile îngroșate indică diferențele dintre fig. 6 și fig. 7.

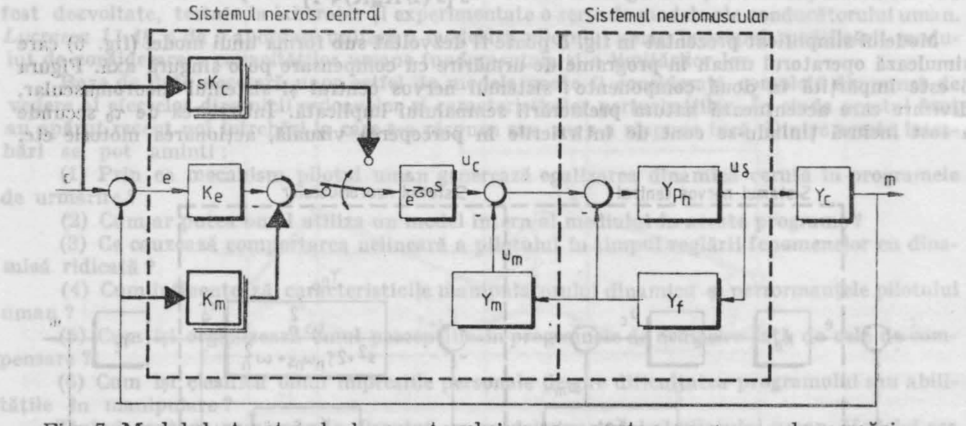


Fig. 7. Modelul structural al operatorului uman pentru programe de urmărire.

Modelul din fig. 7 este alcătuit făcînd următoarele presupuneri : permițînd constantelor de timp T_1 și T_2 să aibă o gamă suficient de largă de valori, funcția de transfer m/u_c poate fi considerată a fi un integrator pentru un domeniu larg de frecvență sub frecvența de tăiere. Mărimea acestor constante poate fi aleasă astfel încît căderea de fază la frecvențe joase, care în mod normal este asociată cu dinamica compensatorie a operatorului uman, să dispară. În plus, acceptînd izomorfismul modelului, dacă operatorul poate sesiza viteza de variație a intrării, dinamica elementului comandat poate fi efectiv inversată la frecvențe joase, permițînd ca dc (t)/dt să fie intrarea primară a buclei din față. Limitarea lărgimii de bandă a elementului comandat inversat nu este prea importantă, deoarece operatorul uman selectează în general frecvența de tăiere sub lărgimea de bandă a intrării, dacă este posibil (McRuer și Krendel, 1974).

Utilizînd modelul structural din fig. 6 poate fi evaluată o medie care reflectă dificultățile programului sau calitățile în manipulare.

$$\sigma_{u_m} = \sqrt{\frac{1}{\pi}\int_0^\infty \left|\frac{u_m}{c}(j\omega)\right|^2 \Phi_{cc}(\omega)\,d\omega} \qquad (7)$$

Mărimea lui σ_{u_m} variază cu aproape două ordine de mărime de la cele mai ușor la cele mai greu de manevrat elemente comandate.

Lucrarea 11.4/C3 propune studierea componentelor dinamice ale sistemelor de conducere a autovehiculelor în scopul asigurării siguranței traficului. Caracteristicile dinamicii autovehiculelor se schimbă brusc cînd automobilul trece de la o porțiune uscată la una umedă a drumului. În plus, la mersul în curbă, sistemului i se adaugă în plus o forță laterală și un moment. Comportările adaptive ale sistemelor de conducere a autovehiculelor în aceste situații critice

sînt studiate folosind pentru experimentare un simulator de autovehicul. Pentru explicarea comportărilor adaptive s-a folosit un nou model matematic pentru conducătorul de autovehicul, model pentru a cărui realizare a fost utilizată teoria reglării adaptive cu model etalon şi a reglării predictive. În fig. 8 este prezentată schema bloc a modelului de reglare a sistemului de conducere a autovehiculului.

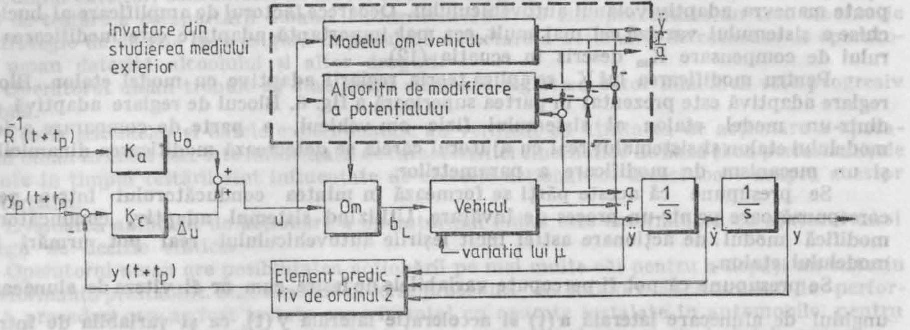


Fig. 8. Modelul de reglare adaptivă a sistemului de conducere a autovehiculului.

Nivelul de reglare predictivă. Funcţiile conducătorilor de autovehicule sînt perceperea variabilelor de ieşire ale autovehiculelor, compensarea prin reacţie pozitivă sau negativă şi întîrzierile datorate sistemului neuromuscular.

(a) *Predicţia de ordinul 2.* Conducătorul poate prevedea devierea viitoare a autovehiculului ţinînd cont de forma curbei. Conducătorii experimentaţi pot prevedea devierea viitoare printr-o lege de predicţie de ordinul 2.

$$y\,(t+t_p)=y\,(t)+t_p\dot{y}\,(t)+\frac{1}{2}\,t_p^2\,\ddot{y}\,(t) \tag{8}$$

unde t_p este timpul de predicţie.

(b) *Compensarea.* Conducătorii pot compensa anticipat unghiul volanului prin reacţia pozitivă realizată prin perceperea razei viitoare a curbei $R(t+t_p)$ chiar cînd raza variază

$$u_0\,(t)=K_a\cdot R\,(t+t_p)^{-1} \tag{9}$$

unde K_a este o constantă oarecare sau o funcţie. Presupunînd că autovehiculul merge pe o curbă circulară, R este constantă şi u_0 este o valoare staţionară. Dacă există o mică deviaţie faţă de punctul prezis, unghiul volanului este compensat folosindu-se această deviaţie drept reacţie negativă

$$\Delta u\,(t)=K_c\,(y_d\,(t+t_p)-y\,(t+t_p)) \tag{10}$$

unde $y_d\,(t+t_p)$ este referinţa în punctul prezis pe curbă. Valoarea totală a unghiului volanului este suma dintre

$$u\,(t)=u_0+\Delta u\,(t) \tag{11}$$

(c) *Sistemul neuromuscular.* Datorită întîrzierii introduse de sistemul neuromuscular, valoarea reală a unghiului volanului nu poate fi realizată utilizînd ecuaţia (11). Se poate presupune că sistemul neuromuscular constă dintr-o întîrziere de ordinul întîi şi un timp mort, după cum urmează

$$b_L\,(t)+t_1\dot{b}_L\,(t)=K_m\,(t)\cdot u\,(t-t_d) \tag{12}$$

unde $b_L\,(t)$ este unghiul real al volanului, t_1 este constantă de timp a filtrului de ordinul întîi, t_d este timpul mort şi K_m este egal cu 1 dacă nu se face adaptare.

Nivelul de reglare adaptivă. După cum s-a menţionat anterior, amplificarea buclei deschise a sistemului variază semnificativ la modificarea coeficientului de frecare şi mai ales la adăugarea unei forţe laterale şi a unui moment datorite mersului autovehiculului pe o curbă cu suprafaţă umedă. Ca urmare, autovehiculul deviază foarte mult spre exteriorul curbei, dacă nu se face sau nu se face suficient o compensare prin reacţie negativă.

Ţinînd cont de studiile experimentale efectuate folosind un simulator, conducătorul poate manevra adaptiv volanul autovehiculului. Deoarece factorul de amplificare al buclei deschise a sistemului variază cel mai mult, cea mai importantă adaptare este modificarea factorului de compensare K_m descris în ecuaţia (12).

Pentru modificarea lui K_m se aplică teoria reglării adaptive cu model etalon. Blocul de reglare adaptivă este prezentat în partea superioară a fig. 8. Blocul de reglare adaptivă constă dintr-un model etalon al sistemului fizic om-vehicul, o parte de comparare a ieşirilor modelului etalon şi sistemului real cu ajutorul căreia se detectează modificarea dinamicii reale, şi un mecanism de modificare a parametrilor.

Se presupune că aceste părţi se formează în mintea conducătorului într-o perioadă corespunzătoare printr-un proces de învăţare. Utilizînd sistemul adaptiv, conducătorul îşi modifică modul de acţionare astfel încît ieşirile autovehiculului real pot urmări ieşirile modelului etalon.

Se presupune că pot fi percepute variabilele de ieşire, cum ar fi viteza de alunecare r(t), unghiul de alunecare laterală a (t) şi acceleraţie laterală y (t), ca şi variabila de intrare în sistemul neuromuscular u (t).

(a) *Viteza de alunecare şi unghiul de alunecare laterală.* În blocul de reglare adaptivă din fig. 8, erorile vitezei de alunecare şi unghiului de alunecare laterală

$$\begin{aligned} \Delta a\,(t) &= a\,(t) - \underline{a}\,(t) \\ \Delta r\,(t) &= r\,(t) - \underline{r}\,(t) \end{aligned} \tag{13}$$

sînt detectate într-o parte de comparare. Folosind teoria stabilităţii Liapunov se poate găsi un mecanism posibil pentru modificarea factorului de compensare K_m

$$K_m\,(t) = -(C_p + C_l/s)\,(h_r\,(\Delta r\,(t) + h_a \cdot \Delta a\,(t)) \cdot u\,(t)) \tag{14}$$

unde integrarea este esenţială pentru a obţine adaptarea stabilă a sistemului nelinear cînd factorul de amplificare al modelului dinamic al autovehiculului scade sub o anumită valoare pe drum umed, factorul de compensare K_m (t) devine convergent spre o anumită valoare după un oarecare timp de adaptare.

(b) *Acceleraţia laterală.* Dacă unghiul de alunecare laterală este foarte mic, dinamica răspunsului vitezei de alunecare este aproape similară cu cea a acceleraţiei laterale. În condiţii experimentale, răspunsul unghiului de alunecare laterală este mult mai mic decît cel a vitezei de alunecare. Se selectează de aceea acceleraţia laterală y (t) pentru a detecta variaţia parametrilor autovehiculului. Eroarea acceleraţiei laterale

$$\Delta \ddot{\bar{y}}\,(t) = \ddot{y}\,(t) - \underline{\bar{y}}\,(t) \tag{15}$$

este sesizată într-un bloc de comparare din fig. 8. Legea posibilă de adaptare care utilizează această informaţie este

$$K_m\,(t) = -(C_p + C_l/s)\,(h_y \Delta \bar{y}\,(t) \cdot u\,(t)) \tag{16}$$

Parametrii ecuaţiilor (14) şi (16) sînt identificaţi experimental folosindu-se un simulator.

Analizînd cu ajutorul simulatorului un accident datorit parcurgerii de către autovehicul a unei suprafeţe umede în curbă, posibilitatea producerii accidentelor, inclusiv a cazurilor periculoase, cînd vehiculul de deplasează în apropierea axei drumului este de aproximativ 60—80 %, în timp ce spre marginile laterale este de aproximativ 20—40 %. Acest fapt nu depinde de direcţia curbei. În lucrare sînt prezentate rezultatele experimentale şi cele obţinute simulîndu-se pe calculator comportarea modelului.

Viteza vehiculului şi raza curbei sînt parametri foarte importanţi la mersul în curbă, deoarece ecuaţiile dinamicii vehiculului depind de viteza lui, iar forţa laterală şi momentul depind de viteza lui şi forţa centrifugă.

Conform rezultatelor experimentale şi simulării pe calculator, rularea cu 80 km/oră este periculoasă. Adaptarea modului de conducere a autovehiculului este efectuată cu succes la 60 km/oră, dar este periculoasă la viteze mai mari de 70 km/oră.

Pentru a detecta acţionarea defectuoasă a operatorului uman afectat de efectele nocive ale mediului exterior, consumului de alcool sau alte droguri au fost dezvoltate şi validate dispozitive de testare a comportării umane. În *lucrarea 11.4/C2* sînt prezentate un test cibernetic şi o strategie de decizie care asigură mijlocul de detectare a acţionării defectuoase a operatorului uman datorită alcoolului şi altor droguri.

Operatorul uman trebuie să stabilizeze elemente reglate a căror dinamică este progresiv instabilă.

Teoria sistemelor şi datele experimentale au verificat că utilitatea de acţionare a operatorului uman în acest test este influenţată de caracteristici cibernetice de bază şi că performanţele obţinute în timpul testării sînt influenţate de efectele acţionărilor defectuoase asupra acestor caracteristici.

Evaluarea abilităţii de acţionare a operatorului uman este determinată cu ajutorul unei strategii de decizie statice.

Operatorul uman are posibilitatea acţionării pe mai multe căi pentru a depăşi un criteriu de performanţă prestabilit. Sînt descrise diverse proeeduri de stabilire a criteriului de performanţă, proceduri care au fost testate experimental cu aparate instalate în automobile pentru a-i descuraja pe conducătorii de vehicule de a consuma alcool atunci cînd sînt la volan.

Testul implică două componente, şi anume o componentă de comandă şi o strategie de identificare. Componenta de comandă, denumită componentă de urmărire în regim critic (CTT), a fost utilizată pentru prima dată ca o metodă de detectare a acţionării defectuoase de către compania General Motors în anul 1973.

Teoria de decizie statică pentru optimizarea strategiei de detectare a fost dezvoltată şi validată în teste de laborator.

Componenta de urmărire in regim critic (CTT). Dinamica componentei constă dintr-un element de comandă instabil şi un circuit de autoacordare. În fig. 9 a şi b sînt prezentate elementele componente ale unui CTT şi analiza stabilităţii locului rădăcinilor. Nu este necesară o intrare, deoarece zgomotul este suficient pentru a perturba sistemul.

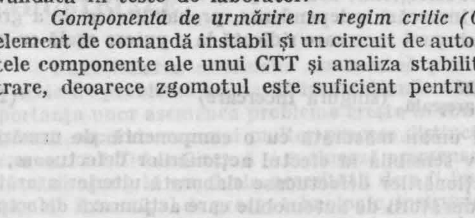


Fig. 9 a. Schema bloc.

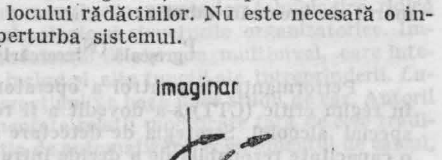


Fig. 9 b. Locul rădăcinilor.

Rădăcina instabilă λ are iniţial o valoare mică. Subiectul începe să execute testul, instabilitatea unităţii creşte (rădăcina se mută în planul din dreapta). Timpul total al testului este de 30 secunde.

După cum se vede din fig. 9, performanţele unui CTT pentru un subiect depind de o constantă de timp vizual/motor (τ_e), de o constantă de amplificare K_p şi de zgomotul intern sau de acţiuni de comandă aleatoare. Constanta de timp de întîrziere determină forma locului rădăcinilor, în timp ce constanta de amplificare K_p determină punctul curent pe locul rădăcimilor.

Creşterea instabilităţii (λ) translatează locul rădăcinilor la dreapta sau în direcţia instabilităţii. Factorul de amplificare determină primele două rădăcini ale buclei închise (constanta

de timp de întîrziere dă un număr infinit de rădăcini, dar perechea de la frecvenţa cea mai mică dictează caracteristicile de stabilitate).

Strategia optimă este de a da o valoare constantei K_p, astfel încît polii buclei închise să se afle pe axa imaginară. Acţionările defectuoase pot să afecteze capacitatea de comandă a operatorului uman pe 3 căi : 1) prin creşterea timpului de întîrziere vizual/motor ; 2) interferenţele în reglările precise ale lui K_p ; 3) creşterea zgomotului. Orice combinaţie a acestor 3 efecte ale acţionărilor defectuoase pot să reducă raportul obţinut de un CTT pentru λ_c.

Numeroase studii au demonstrat efectele alcoolului asupra lui λ_c.

Strategia de detectare a acţionărilor defectuoase (IDS). Obiectivul unui IDS este de a mări şansa de a detecta acţionările defectuoase ale unui operator cu un număr minim de CTT. Optimizarea şi dezvoltarea unui IDS începe cu analiza proprietăţilor statistice ale performanţelor unui CTT (λ_c).

Bazat pe o analiză statistică a numeroase date, IDS a stabilit următoarele cerinţe :

1) diferenţele importante de performanţă între operatori necesită criterii de performanţă individualizate ;

2) variaţia înregistrărilor performanţelor stabile şi variaţia performanţelor încercărilor independente permit utilizarea unei strategii simple de eşantionare multiplă ;

3) îmbunătăţirea priceperii în termen lung ar necesita proceduri de eşantionare şi înnoire periodică a criteriului de performanţă.

Caracteristici statistice importante de performanţă raportate la elaborarea unui IDS pot fi ilustrate cu funcţii de distribuţie cumulative. Distribuţiile sînt normalizate şi mediate pe un număr mare de subiecţi şi reprezentate grafic pe formulare speciale (o distribuţie gaussiană apare reprezentată ca o linie dreaptă). Datele sînt distribuite normal pe o gamă largă şi efectele alcoolului sînt indicate clar.

Cerinţa de bază a unui IDS este că performanţa eşantionată a subiecţilor trebuie să depăşească un nivel de trecere prestabilit. Au fost analizate mai multe strategii de eşantionare cu baze de date vechi şi prin diverse criterii a fost selecţionat doar un prag din mai multe încercări admise. Cu această strategie şi presupunînd încercări independente, probabilitatea de a greşi testul este probabilitatea de a greşi pentru o singură încercare ridicată la o putere egală cu numărul de încercări admise :

$$P_{\text{greşeală}}(N_{\text{încercări}}) = P^{N}_{\text{greşeală}}\ (\text{singură încercare}) \tag{17}$$

Performanţa de control a operatorului uman măsurată cu o componentă de urmărire în regim critic (CTT) s-a dovedit a fi relativ sensibilă la efectul acţionărilor defectuoase, în special alcoolul. Strategia de detectare a acţionărilor defectuoase elaborată ulterior a arătat o capacitate rezonabilă de a decide între conducătorii de automobile care acţionează defectuos şi cei care acţionează corect. Combinaţia CTT/IDS s-a dovedit a lucra în mod fiabil o perioadă de timp extinsă într-un mediu de lucru în care utilizatorii nu erau în mod necesar motivaţi să coopereze cu funcţia de control. Au fost elaborate proceduri pentru instruirea operatorilor şi utilizatorilor şi pentru fixarea unor praguri de trecere bazate pe capacitatea individuală de performanţă. CTT/IDS poate fi utilizat ca un dispozitiv cibernetic de filtrare şi în alte servicii ca urmărirea zilnică a operatorilor de vehicule comerciale sau guvernamentale, urmărirea operatorilor din procesele industriale sau centrale electrice.

Strategia de detectare a acţionărilor defectuoase (IDS) poate fi utilizată şi în alte activităţi cibernetice care se dovedesc a fi sensibile la alte aspecte sau la acţionări defectuoase ale operatorului uman.

Cercetări recente s-au făcut în domeniul telecomenzii vehiculelor roboţi şi a manipulatoarelor, după cum se arată şi în *lucrarea 11.4/C6*. Aceste cercetări, făcute în cadrul Laboratorului sistemelor om-maşină — MIT, analizează utilizarea calculatorului în ajutorul operatorului uman precum şi afişările utilizate în cazul manipulatoarelor utilizate în oceane la mare adîncime. Aceste sisteme sînt utilizate pentru a înlocui scafandri umani în cercetările la mare adîncime şi de mari proporţii şi la operaţiile care au tendinţă de a fi foarte riscante şi costisitoare.

Operatorul uman îndeplineşte funcţia de controlor supervizor, comunicînd cu calculatorul care este în legătură cu un vehicul/manipulator îndepărtat ; calculatorul interpretează şi execută comenzi cu ajutorul unor traductoare proprii ale vehiculelor şi manipulatoarelor.

Multe firme au produs vehicule utilizate numai pentru prospectare, dotate cu camere video sau vehicule utilizate atît pentru prospectare, cît şi pentru lucru, dotate cu camere video, braţe şi dispozitive mecanice. Pentru conducerea acestor vehicule de la suprafaţa vapoarelor,

operatorul uman urmăreşte un monitor video şi comandă vehiculele şi/sau manipulează braţe. O parte din sisteme au comandă master/slave cu multe grade de libertate. Dîndu-se vehiculelor cîteva grade de libertate şi permiţînd operatorului uman să comunice cu vehiculele intermitent printr-o legătură de comunicaţie acustică (lăţime de bandă joasă), operatorul uman încetează să mai fie un „controlor în buclă", deservind în schimb un operator supervizor de la distanţă care transmite instrucţiuni la calculator. Calculatorul execută task-urile şi închide bucla, utilizînd senzorii proprii ai vehiculelor.

Pentru operatorul supervizor pot fi identificate 5 funcţii:
(1) planificarea misiunilor la fiecare nouă actualizare a instrucţiunilor către robot;
(2) comunicarea acestor instrucţiuni către calculator;
(3) afişarea de supraveghere care să asigure că programele se execută corect;
(4) intervenţia pentru actualizarea programului sau pentru a prelua comanda în caz de avarie;
(5) folosirea experienţei.

BIBLIOGRAFIE

Hess, R.A. (1981) **Pursuit tracking and higher levels of skill development in the human pilot.** IEEE Trans. on Sys., Man, and Cybernetics, SMC-11, 262—273.

Mc Ruer, D.T., *şi E. Krendel* (1974) **Methematical models of human pilot behavior.** AGARDograph No 188.

SUPRAVEGHEREA PROCESELOR ÎN SISTEMELE OM-MAŞINĂ INDUSTRIALE

Funcţionarea sistemelor om-maşină pentru supravegherea proceselor tehnologice ridică o serie de probleme legate de interacţiunile dintre tehnologie şi structurile organizatorice. Importanţa unor asemenea probleme creşte în condiţiile folosirii de sisteme multinivel, care integrează supervizarea mai multor procese distincte şi includ şi alte funcţii ale întreprinderii. *Lucrarea 11.4/D2* analizează şi evaluează asemenea interacţiuni pe baza unor studii de caz. Autorii lucrării ajung la concluzia necesităţii de a fi implementate relaţii modificate faţă de cele tradiţionale între tipul de proces tehnologic, instrumentaţia de automatizare, echipamentul de calcul, configurarea camerelor de comandă şi a consolelor operator, activităţile de supervizare şi structurile organizatorice. Autorii se referă numai la procese continue şi semicontinue, dar se poate considera că rezultatele studiului lor sînt utile şi în cazul proceselor discontinue (exemplu construcţia de maşini).

În practica actuală, operatorii lucrînd în echipe, în proiectarea unui sistem om-maşină este necesar să fie luate în considerare 2 etape în repartizarea de sarcini:

— repartizarea de sarcini sistem între maşină şi om;

— repartizare de sarcini „om" între operatori, cu divizarea corespunzătoare a responsabilităţilor şi stabilirea unor reguli de comunicare, altfel spus determinarea unei structuri organizatorice.

Lucrarea menţionează modul tradiţional de abordare, în care organizarea muncii operatorilor este influenţată univoc de următorii factori:

— tehnologia utilizată în procesul supervizat;

— tradiţia de organizare din întreprindere;

— configurarea camerelor de comandă şi a consolelor şi comunicaţia om-maşină, dependente univoc la rîndul lor de echipamentul de calcul şi instrumentaţia de automatizare disponibile comercial.

În lucrare sînt menţionate preocupările în acest domeniu ale altor specialişti. Există, în opinia autorilor lucrării, 4 categorii de asemenea preocupări:

— Experimente de laborator cu sisteme om-maşină, cu luarea în considerare a unor variabile organizatorice. Este menţionată o lucrare a lui Terano, Murayama şi Akiyama (1983) care, experimentînd sisteme om-maşină cu 1 şi 2 operatori, au ajuns la concluzia că fiabilitatea unui operator scade atunci cînd el cooperează cu un al doilea operator, faţă de cazul concurenţei sau al muncii individuale;

— Încercări de a optimiza organizarea muncii operatorilor în condiţii de tehnologie impusă. Este menţionată o lucrare a lui Bailey (1983), care, în opoziţie cu Taylor sau cu Weber, consideră că pentru creşterea eficienţei muncii este necesară o creştere a gradului de satisfacţie. O limită a cercetărilor lui Bailey ar consta în slaba utilizare a gradelor tehnologice de libertate susţin autorii lucrării 11.4/D2 ;

— Studii descriptive ale interacţiunilor dintre tehnologie şi organizare. Sînt amintite analizele, de acum clasice, ale lui Woodward (1965) şi Perrow (1967), punîndu-li-se în evidenţă limitele (Perrow ajunge la o soluţie weberiană pentru cazul tehnologiilor de rutină). Este menţionată de asemenea o lucrare, mult mai recentă, avîndu-l ca principal autor pe Ekkers (1980), care se ocupă de relaţia dintre gradul de satisfacţie şi gradul de automatizare, cei doi factori fiind, de la un anumit nivel de automatizare, contradictorii ;

— Experimente de teren cu adaptarea mutuală a tehnologiei şi organizării. Este citat aici Bibby (1975), cu un exemplu de minimizare a deranjamentelor într-o oţelărie, minimizare obţinută nu prin îmbunătăţirea instrumentaţiei, ci prin schimbări organizatorice : fiecare operator răspunde nu numai de unitatea proprie, ci şi de influenţa asupra unităţii din aval. Pentru ca noua responsabilitate să poată fi îndeplinită efectiv, operatorului i se furnizează informaţii din aval.

Autorii lucrării 11.4/D2 ne înfăţişează apoi propriile lor preocupări. Ei pornesc de la definirea funcţiilor operaţionale şi a relaţiilor dintre ele, sau, în exprimarea lor, a cadrului funcţiilor operaţionale. Prin funcţii operaţionale autorii înţeleg categorii de sarcini atribuite omului şi maşinii pentru asigurarea siguranţei în exploatare, a nivelului calitativ şi cantitativ al producţiei, al eficienţei (definiţia este reluată dintr-un articol anterior al unuia din autori, Rijnsdorp).

Cadrul funcţiilor operaţionale constituie un sistem complex structurat pe mai multe nivele :

— nivelul de bază, reprezentat de bucla : proces (cu unităţi de corecţie asociate) — măsurare-automatizare (protecţie, automatizare secvenţială, reglare) — proces ;

— nivelul de compensare/corecţie, reprezentat de legătura măsurare-evaluare stare proces-diagnoză-compensare sau corecţie a proastei funcţionări ;

— nivelul de conducere adaptivă, reprezentat de legătura măsurare-evaluare stare proces-estimare a calităţii şi eficienţei-conducere adaptivă (prin ajustare de temporizări, de scheme de comandă, de mărimi de referinţă, etc.) — automatizare ;

— nivelul de optimizare, reprezentat de legătura măsurare actualizare model-optimizare (prin reţete, valori critice, valori impuse)-conducere adaptivă. Menţionăm că autorii au în vedere optimizarea atît a procesului, cît şi globală, a secţiei/întreprinderii. De la secţie/întreprindere se transmit spre proces valori critice, debite etc. ;

— nivelul întreprindere, reprezentat de legătura măsurare-informare operativă-programare producţie şi gestiune stocuri-optimizare.

Referitor la nivelul întreprindere, autorii menţionează că legătura informare operativă-programare producţie şi gestiune stocuri se realizează atît direct, cît şi prin intermediul funcţiei de planificare. Funcţiile de planificare, programare producţie şi gestiune stocuri interacţionează cu alte funcţii la nivelul întreprindere (livrări, întreţinere etc.), care la rîndul lor pot fi alimentate cu date de către informarea operativă. Existenţa acestor legături la nivelul întreprindere îi conduce pe autori (care îl citează în acest sens pe Forrester) la afirmarea necesităţii unui sistem integrat de conducere a întreprinderii, a cărui filozofie să fie bazată pe triada beneficiari-producţie-furnizori, sistem care să fie alimentat de către informarea operativă.

Studierea acestui cadru complex al funcţiilor operaţionale îi conduce pe autori la concluzia că maximizarea gradului de automatizare nu este în mod necesar echivalentă cu optimizarea conducerii procesului. Funcţiile operaţionale vitale apar a fi măsurarea şi evaluarea stării procesului. Un grad ridicat de automatizare reduce activitatea operatorului în principal la monitorizare. O asemenea situaţie (sînt citate, în sprijinul acestor afirmaţii, lucrări de van Droffelar, 1975 ; Ekkers, 1980 ; şi Bainbridge, 1983) este o sursă de stress, de degradare a satisfacţiei în muncă şi este psihologic nepotrivită pentru comportarea în situaţii neprevăzute. Şi autorii conchid, „nu este întotdeauna recomandabil să ţinteşti spre cel mai înalt grad de automatizare posibil".

În continuarea studiului lor autorii se ocupă de funcţia de evaluare a stării procesului, funcţie cheie în cadrul funcţiilor operaţionale. „Cu instrumentaţia modernă", spun ei, „informaţia este prezentată în camera de comandă pe dispozitive de afişare pentru a fi monitorizată şi interpretată de către operatori. Totuşi, în unele sectoare industriale, această informaţie

trebuie suplimentată prin observarea directă a condițiilor, de exemplu la benzi transportoare, uscătoare, centrifuge etc.".

Autorii disting 3 cazuri pe care le studiază separat:

— starea poate fi evaluată prin afișarea pe consola operator, eventual cu informații suplimentare obținute prin televiziune în circuit închis;

— starea poate fi evaluată numai parțial prin afișarea pe consola operator, fiind necesară și o supraveghere locală;

— evaluarea stării necesită supravegherea locală în timpul unor situații frecvente de porniri, opriri, purjări etc., dar poate fi monitorizată în camera de comandă în situația normală.

Primul caz se întîlnește în procese în care se prelucrează mase de lichid sau gaz (exemplu, petrochimia), al doilea caz se întîlnește în procese în care se prelucrează materiale solide sub formă pulverulentă sau granulară, iar al treilea caz se întîlnește în activități de deplasare/stocare. Cele 3 cazuri sînt luate în discuție în legătură cu activitățile operatorilor asociați cu console-operator și cu organizarea echipelor de operatori.

În primul caz, arată autorii, evaluarea stării este efectuată de operatori centrali, asistați însă de operatori de teren pentru relativ rarele acțiuni de comutare, pentru cooperarea cu echipele de întreținere și pentru supravegherea generală a echipamentelor. Procesul de prelucrare supravegheat fiind alcătuit din mai multe „unități-proces", acestea sînt alocate uneia sau mai multor console, fiecare cu unul sau mai mulți operatori centrali, asistați de operatori de teren.

Din punct de vedere al organizării muncii operatorilor, autorii arată că trebuie rezolvate următoarele probleme:

— cîte unități-proces să se aloce fiecărui operator;

— cum să se mențină în munca operatorilor centrali un grad de încărcare relativ constant în timp.

Referitor la prima problemă, se obișnuiește ca volumul de muncă al unui operator central să fie exprimat în număr de bucle de reglare supravegheate. În cazul sistemelor automate moderne (în cascadă, multinivel și/sau care folosesc algoritmi multivariabili) această abordare este depășită, susțin autorii, care propun exprimarea volumului de muncă în organe de execuție supravegheate. Sînt menționate în context două aspecte care influențează volumul de muncă al unui operator:

— munca unui operator central este încărcată cu un volum important de comunicații, datorită contactelor frecvente cu operatorii de teren, alți operatori centrali, superiori, tehnologi, tehnicieni de întreținere;

— este o mare diferență între operarea tradițională, în care se urmărește menținerea unor condiții prestabilite și operarea modernă, în care valorile impuse se modifică frecvent, iar limitele de încadrare ale valorilor critice sînt foarte strînse.

Referitor la problema menținerii în munca operatorilor centrali a unui grad de încărcare relativ constant în timp (ceea ce înseamnă evitarea diferenței de încărcare situație normală-situație de urgență), sînt propuse următoarele soluții:

— includerea de sarcini de tip „background" în munca operatorilor centrali, care să poată fi neglijate pe durata situațiilor de urgență;

— delegarea către operatorul central a responsabilității de coordonare a activității operatorilor de teren;

— asistarea operatorului central pe durata situațiilor de urgență de către unul din operatorii de teren sau de către un alt operator central (ceea ce implică o cameră de comandă centrală, care să concentreze toți operatorii centrali). Pentru ca asistarea să fie de calitate este bine să se asigure o rotație periodică a operatorilor centrali între ei (deci de la un grup de unități proces la altul), precum și a operatorilor centrali cu operatorii de teren.

Autorii consideră că aspectele organizatorice au implicații asupra proiectării optimizate a camerei de comandă și a consolelor (forma și mărimea consolei, forma și mărimea caracterelor afișate etc.).

Este luat apoi în discuție cazul proceselor în care se prelucrează materiale solide sub formă pulverulentă sau granulară, la care sesizarea stării se face parțial prin afișare pe consolă operator, în camera de comandă, și parțial prin supraveghere directă. Aici autorii consideră că o structură organizatorică cu 2 operatori (central și de teren) pentru un grup de unități-proces este nepotrivită, implicînd un volum important de comunicații, cauză de erori, neînțelegeri, frustrare. Structura organizatorică optimă în acest caz presupune pentru un grup de unități proces un singur operator circulînd între camera de comandă și teren.

Necesitatea unei camere de comandă centrale, care să concentreze toţi operatorii este argumentată astfel:

— permite cooperarea între operatori pentru optimizare globală;

— oferă posibilitatea asistenţei mutuale.

În cazul în care distanţa dintre camera de comandă centrală şi proces ar fi prea mare, se impun camere de comandă locale, dar, susţin autorii, care să concentreze totuşi cel puţin 2 operatori (supravegherea a 2 grupe de unităţi-proces, deci).

Autorii iau apoi în discuţie cazul proceselor în care sînt frecvente porniri, opriri, purjări etc. (exemplu, activităţi de stocare/deplasare). În acest gen de procese este de efectuat un mare volum de supraveghere de teren, camera de comandă nefiind necesară.

Dacă însă procesul se desfăşoară pe o suprafaţă foarte mare, camera de comandă devine necesară, pentru a putea supraveghea procesul în ansamblu.

După această discutare amănunţită a funcţiei de evaluare stare proces, autorii se referă la nivelele superioare ale cadrului funcţiilor operaţionale. Ei arată că în prezent există o puternică tendinţă de integrare a proceselor individuale, prin schimbul de energie şi materiale. Această integrare impune o bună calitate a comunicării dintre operatori, preferabil prin contacte directe vizuale şi auditive (în special în situaţii de urgenţă), ceea ce este posibil numai în camere de comandă combinate şi are implicaţii în proiectarea optimizată a consolelor (de exemplu, apare necesar ca dimensiunile consolelor să permită contacte vizuale între operatori).

În ceea ce priveşte nivelul de optimizare, autorii menţionează un experiment dintr-o întreprindere chimică, care a dus la concluzia existenţei unui set optimal de valori discrete ale debitelor de agenţi energetici. Operarea este optimă cînd echipamentele energetice asigură numai aceste valori discrete, în funcţie de necesităţile momentane ale întreprinderii. Cînd apare o mare cerere de energie, se impune funcţionarea unor echipamente (turbine, cazane, etc.) la capacitatea maximă, cînd există o abundenţă de energie, echipamentele respective ar trebui să fie deconectate. Conducerea optimală este diferită de cea tradiţională, iar echipei de operatori îi revin deci sarcini suplimentare: supravegherea simultană a unor algoritmi de comandă divizaţi pe operatori, capacitatea de a răspunde rapid la schimbările în necesarul de agenţi energetici. Aceasta impune adoptarea unor soluţii de software „prietenos" pentru console: posibilităţi eficiente de dialog, scheme adecvate etc.

Trecînd la nivelul întreprindere, autorii pledează pentru un sistem integrat de conducere, implementat pe o reţea de calculatoare, deoarece, spun ei, „fără o reţea de calculatoare, o asemenea integrare ar necesita strîngerea tuturor persoanelor cu atribuţii de programarea producţiei într-o echipă bine îmbinată, ceea ce ar interfera cu restricţiile de timp de comunicare cu personalul operativ. O reţea bine proiectată permite fiecăruia să rămînă în sectorul propriu."

În concluzie, abordarea preconizată de autori a interacţiunilor tehnologie-structură organizatorică diferă faţă de abordarea tradiţională, menţionată la începutul lucrării.

Organizarea muncii operatorilor nu trebuie să depindă de tradiţia existentă în întreprindere, ci invers, într-o oarecare măsură organizarea muncii operatorilor ar trebui să influenţeze tradiţia.

Organizarea muncii operatorilor trebuie să impună soluţia de proiectare a camerei de comandă şi a consolei şi nu invers. Aceasta impune minimizarea influenţei echipamentelor de calcul şi instrumentaţiei de automatizare disponibile comercial asupra camerei de comandă şi consolei. Sînt de dorit soluţii de echipamente şi programe „pe măsură", care să rezulte din soluţia de proiectare a camerei de comandă şi a consolei.

Calitatea informaţiei furnizate pe consolă determină aşadar într-o măsură însemnată succesul unui sistem om-maşină. Aceasta este şi opinia autorilor *lucrării 11.4/D4* Tsuchiya de la Toshiba Corporation, Tokio şi Uchida, care prezintă sistemele „prietenoase" (operator friendly) folosite în termocentrale din Japonia.

Implementarea în termocentralele japoneze a sistemelor cu calculator reprezintă un succes incontestabil, datorită următorilor factori:

— necesitatea resimţită de utilizatori de reducere a cheltuielilor de exploatare şi de minimizare a duratei procedurii zilnice de conectare/deconectare;

— gradul înalt de fiabilitate al sistemelor cu calculatoare de proces, observat în exploatare;

— volumul minim de probleme de interfaţă între beneficiari, furnizorii echipamentului energetic şi cei ai echipamentului de calcul, avînd în vedere că uzual calculatorul este inclus în furnitura complexă;

— legătura funcţională care există între calculator şi automatizările centralei ;
— existenţa unui serios suport software de testare ;
— creşterea în timp a complexităţii acestor sisteme, plecînd de la un set minim de funcţii.

Ultimul factor apare, în opinia autorilor, ca fiind cel mai important. Evoluţia sistemelor se prezintă astfel :

— comanda pornirii pentru turbine (1968) ;
— comanda conectării pentru cazane (1971) ;
— comanda conectării/deconectării pentru instalaţiile auxiliare (1976) ;
— comanda deconectării întregii centrale (1976) ;
— operarea în situaţie normală (1980) ;
— operarea în situaţii de urgenţă (1980) ;
— aplicarea metodelor moderne de conducere automată. Autorii menţionează în acest sens un articol al lui Uchida (1981), în care se prezintă sistemul de la termocentrala Buzen, unde metodele moderne de conducere automată au fost folosite la reglarea temperaturii din cazan.

În Japonia strategia de asigurare a necesităţilor de energie electrică acordă centralelor nucleare rolul de centrale de bază, termocentralelor rezervîndu-li-se rolul de centrale de vîrf. Pentru acest motiv s-au dezvoltat sisteme om-maşină care să asigure procedura de conectare/deconectare zilnică a termocentralei. Aceste sisteme îşi propun 4 obiective principale :

— conectare/deconectare sigură, fără şocuri, fiabilă. Autorii îl menţionează pe Kato (1978) care într-un articol prezintă pachetul COPOS, destinat acestui obiectiv ;
— reducerea cheltuielilor de exploatare a echipamentelor, Karashima, într-un articol publicat în 1981, descrie implementarea în vederea realizării acestui obiectiv, pe sistemul de la termocentrala Hirono, a unei funcţii de determinare a conectării/deconectării optimale ;
— reducerea duratei conectării/deconectării ;
— reducerea gradului de încărcare în activitatea operatorului la conectare/deconectare.

Ultimelor 2 obiective le sînt dedicate noile sisteme TOSBAC, cu interfaţă „prietenoasă„ (operator friendly), descrise de către autori.

Interfaţa om-maşină a calculatoarelor TOSBAC a evoluat astfel :

— consolă dedicată pentru concetări/deconectări (1968) ;
— display alb-negru (1971) ;
— display color alfanumeric, funcţii de alarmare şi informare (1976) ;
— display color grafic, sistem analogic de alarmare prin voce, funcţii de software „prietenos" (1980). Menţionăm că pentru afişare de mesaje în limba japoneză sînt necesare facilităţi grafice :
— sistem digital de alarmare prin voce, rafinare a funcţiilor de software „prietenos" pentru procedura de conectare/deconectare zilnică a termocentralei (1983).

Se observă corespondenţa cu etapele de evoluţie a funcţiilor atribuite calculatorului. Menţionăm că anii din paranteze indică implementări ale sistemelor în termocentrale.

Pentru a demonstra necesitatea unei interfeţe cît mai „prietenoase", autorii lucrării arată că sistemul nu este conceput în ideea de a opera fără intervenţia omului. Maşina colaborează cu omul asistîndu-se reciproc. De aceea extinderea domeniului pe care îl controlează sistemul trebuie să implice creşterea calităţii informaţiilor oferite operatorului.

Efectuarea unei operaţii de comandă asupra unui echipament se reduce la îndeplinirea a 3 categorii de condiţii (autorii reiau aici concluziile unui studiu al lui Tanaka, 1975) :

— condiţii care trebuie îndeplinite înainte de începerea operaţiei ;
— condiţii care trebuie îndeplinite pe durata efectuării operaţiei ;
— condiţii a căror îndeplinire semnifică încheierea operaţiei.

Efectuarea operaţiilor de conectare/deconectare a unei termocentrale necesită îndeplinirea secvenţială de condiţii din cele 3 categorii, condiţii care se constituie într-o procedură complexă.

Funcţiile de software „prietenos" dezvoltate pe sistemele TOSBAC oferă operatorului posibilitatea de a urmări procedura de conectare/deconectare prin următoarele tipuri de afişări pe display :

— indicarea stării unei operaţii prin afişarea de mesaje de informare asupra îndeplinirii/neîndeplinirii condiţiilor ;
— afişarea de „hărţi logice" ale operaţiilor cu indicarea îndeplinirii/neîndeplinirii condiţiilor ;
— afişarea de scheme sinoptice ale instalaţiilor cu indicarea locului unde s-a produs un deranjament ;
— afişarea de histograme.

Aceste funcţii se adaugă funcţiilor disponibile de mai mult timp pe display (afişări de alarme, raportări la cerere etc.) şi pe consola dedicată pentru conectări/deconectări.

Erorile tipice ale unui operator pot aparţine uneia din următoarele 3 faze (după Oxby, 1982):

— percepţie;
— decizie;
— operare.

Funcţiile prezentate în lucrare corespund fazei „operare". În încheiere, se afirmă că tendinţa de evoluţie a sistemelor om-maşină din termocentralele japoneze este spre faza „decizie". În acest sens, într-un viitor apropiat vor apare sisteme om-maşină bazate pe cunoştinţe.

Asistenţa operatorului în faza „decizie" este un subiect discutat în *lucrarea 11.4/D3*, care prezintă în acest sens un sistem în curs de testare într-un combinat siderurgic din R.D.G. (VEB Maxhütte Unterwellenborn).

Într-un mare număr de procese tehnologice complexe, arată autorii, omul trebuie să ia decizii de comandă pe baza observării procesului şi pe baza propriei sale experienţe şi intuiţii. Aceste decizii sînt evident subiective (problema este dezbătută şi de Brack şi Sokollik, 1982; şi de Förster şi Starke, 1982).

Scopul pe care şi l-au propus autorii lucrării 11.4/D3 este acela de a dezvolta instrumente de asistare a deciziei sub forma unor „unităţi consultative", în special pentru procese cu următoarele caracteristici:

— informaţiile despre proces nu se pot obţine decît inexact sau sub formă de evaluări subiective;
— conexiunile interne din proces nu sînt suficient de cunoscute teoretic pentru elaborarea deciziei;
— procesul este nelinear, cu perturbaţii puternice;
— unele variabile esenţiale ale procesului nu pot fi măsurate, sau pot fi măsurate numai indirect.

Aceste „unităţi consultative" trebuie să dea un suport obiectiv deciziei. În elaborarea propunerilor de decizii, unităţile consultative se bazează pe analiza procesului, pe tehnici de analiza erorilor, pe tehnici de optimizare şi de asemenea pe experienţa cîştigată de colective largi de operatori.

O unitate consultativă îşi îndeplineşte sarcinile în 2 etape:

— recunoaşterea şi clasificarea situaţiei;
— elaborarea propunerii de decizie.

Cele 2 etape pot fi implementate pe unităţi de calcul diferite. În acest sens, una din unităţile consultative prezentate în lucrare este implementată pe 2 calculatoare, denumite calculatorul de analiză proces şi calculatorul consultant (în sensul în care îl vede operatorul), amîndouă fiind subordonate unui calculator de conducere a procesului tehnologic, subordonat la rîndul lui calculatorului central al combinatului.

Sistemul în curs de testare la VEB Maxhütte Unterwellenborn cuprinde 5 unităţi consultative, situate pe lanţul convertizor-turnătorie-laminor. În lucrare este prezentată, succint, strategia de funcţionare a unităţii consultative pentru afinare finală de la convertizor.

Asemenea „unităţi consultative" permit integrarea operatorului cu toate capacităţile sale în sistemul om-maşină şi aduc beneficii economice importante (în cazul prezentat, după o perioadă de 2 ani, beneficiul obţinut a fost de 10 ori mai mare decît totalul cheltuielilor, afirmă autorii).

Cele 3 lucrări prezentate pînă acum (11.4/D2, D3 şi D4) se ocupă de sisteme om-maşină în industrie. Există şi un alt domeniu în care sistemele om-maşină îşi găsesc aplicabilitate, cel al cercetării ştiinţifice, al experimentelor de laborator. Este cazul sistemului prezentat în *lucrarea 11.4/D5*, sistem folosit pentru măsurători în acceleratoarele de protoni.

Acceleratoarele produc particule încărcate, în serii (cicluri de accelerare) la intervale de milisecunde. Pentru determinarea performanţelor acceleratoarelor sînt necesare măsurări repetate asupra fasciculului de particule la momente arbitrare de timp în cadrul ciclului de accelerare.

Instrumentele de măsurat pot fi complet diferite între ele din punct de vedere al principiilor folosite, dar au 2 caracteristici comune: necesităţile de operare şi necesităţile severe de timp real.

Necesităţile de operare se referă la folosirea instrumentelor de măsurat în regim multiutilizator. În timpul fazei delicate de acordare a acceleratorului mai multe persoane pot dori

accesul la același instrument de măsurat în același ciclu de accelerare. Soluția multiplicării instrumentelor de măsurat ar prezenta 2 dezavantaje:

— cost ridicat;

— lipsă de precizie: diferite instrumente prezintă ușoare diferențe între ele (calibrare, derivă amplificator etc.), ceea ce reduce posibilitatea de comparare între măsurări.

Limitele de timp real sînt severe: intervalul minim între 2 măsurări este de 2—3 milisecunde. De menționat că o măsurare impune transferul cuvintelor de comandă și uneori a sute de cuvinte de date.

Existența acestor caracteristici comune a condus la soluția de a separa în fiecare echipament de măsurat partea specifică de partea standard (hardware și software). A rezultat o configurație hardware și software, care a fost denumită MTIM (Multiple Trigger for Measurements), la care se poate conecta un instrument de măsurat, independent de complexitate.

Sistemul MTIM este configurat într-un sertar CAMAC, mai multe asemenea sertare fiind cuplate la magistrala serială CAMAC a unui calculator de proces, conectat la rîndul lui la o rețea, prin care comunică cu 6 console de la care sînt accesate măsurările. Un sertar CAMAC conține componentele hardware ale MTIM (4 module) precum și modulele specifice instrumentului de măsurat respectiv.

Configurația software a MTIM este compusă din:

— un modul rezident în calculatorul de proces (comun pentru sistemele MTIM cuplate);

— un set de module rezidente în memoria RAM a controlorului sertarului (unul din cele 4 module microprocesor TMS 9900 pe 16 biți cu 16 K memorie RAM), la care se adaugă modulele software specifice instrumentului de măsurat respectiv. Modulele rezidente în memoria controlorului sînt încărcate din memoria calculatorului de proces prin magistrala serială.

Sisteme MTIM, menționează în încheiere autorii, sînt în funcțiune la Centrul European de Cercetări Nucleare de aproximativ 2 ani, avînd în sarcină 10 măsurări diferite.

BIBLIOGRAFIE

Bailey, J. (1983). **Job design and work organization**; Prentice Hall.

Bainbridge, L. (1983). **Ironics of Automation**; In: Proc. IFAC/IFIP/IFORS/IEA Conf. on Analysis, Design and Evaluation of Man-Machine Systems; Baden-Baden, Pergamon.

Bibby, K.S. (1975). **Human and organizational problems of production control**; In: Proc. IFAC Workshop, Bad Boll (RKW, Frankfurt).

Brack, G. și *F. Sokollik* (1982). **Die ingenieurtechnischen Aufgaben bei den operativen Lenkungen der Produktion.** Chemische Technik, 34, 2, p. 61—63.

Droffelaar, H. van (1975). **A field study of stress, experienced by operators supervising a highly automated process**; In: IFAC Workshop, Bad Boll (RKW, Frankfurt).

Ekkers, C.L. ș. a. (1980). **Human control tasks** (în limba daneză); NIPG, Leiden.

Förster, D. și *J. Starke* (1982). **Erfahrungen bei der Anwendung von Entscheidungshilfen für die operative Lenkung.** Chemische Technik, 34, 2, p. 64—66.

Kato, N. (1978). **Application of Computer Control System COPOS to Fossil Fired Power Generation Plant.** 3-rd USA — JAPAN Computer Conference Proc., p. 505—509.

Karoshima, N. (1981). **Newly Developed Comprehensiv Automation Technique Applied to Hirono Thermal Power Station**; Proc. of American Power Conference, 43-er Annual Meeting, Chicago, p. 1—44.

Oxby, K. (1982). **A Multi-Level Alarm Information Processing System Applied to Thermal Power Plant.** IFAC/IFIP/IFORS/IEA Conf. on Analysis, Design and Evaluation of Man-Machine Systems Proc., p. 89—94.

Perrow, Ch. (1967). **A framework for comparative analysis of organization**; Am. Sociological Rev. 32, p. 194—208.

Tanaka, S. (1975). **New Concept Software System for Power Generation Plant Computer Control** — COPOS. Proc. of PICA Conference, p. 1—9.

Terono, T., Y. Murayama și *N. Akiyama*, (1983). **Human reliability and safety evaluation of man-machine systems**; In: Proc. IFAC/IFIP/IFORS/IEA Conf. on Analysis, Design and Evaluation of Man-Machine Systems; Baden-Baden, Pergamon.

Uchida, M. (1981). **Totally Computer Automated Control System in a Thermal Power Unit.** Proc. of IFAC 8-th Triennal World Congress, xx-135 — xx-141.

Woodward, J. (1965). **Industrial organization, theory and practice**; Oxford Univ. Press.

SISTEME DECIZIONALE IERARHICE ȘI METODE DE DECIZIE MULTICRITERIALE

Mat. *Ovidiu Gheorghiu*,
Mat. *Tatiana Călin*

I.T.C.I.

SISTEME SOCIO-ECONOMICE

În cadrul *colocviului 11.5* al celui de al IX-lea Congres IFAC, lucrările dedicate sistemelor decizionale ierarhice au fost grupate în două secţiuni. Vom prezenta în continuare prima secţiune, ce cuprinde 4 lucrări, a cincea [11.5/A3], figurînd în cuprins, dar nefiind prezentă printre lucrări.

Vom face, pentru început, observaţia că lucrările secţiunii respective au toate un numitor comun, şi anume substratul socio-economic al ideilor şi rezultatelor prezentate. În altă ordine de idei, titlul secţiunii — Sisteme decizionale ierarhice — exprimă doar parţial caracteristica lucrărilor ce tratează într-adevăr probleme decizionale dar nu întotdeauna privite ierarhizat. Trebuie evidenţiată diversitatea unghiurilor de abordare a problemelor decizionale, articolele prezentînd aspecte teoretice şi practice generalizabile pe o arie largă de aplicaţii. În acest context se remarcă faptul că secţiunea cuprinde o dezvoltare a problemelor metodologice specifice ingineriei sistemelor socio-economice cu referiri la situaţii concrete din China [11.5/A2], precum şi o abordare foarte riguroasă a problemelor de modelare a sistemelor socio-economice şi a politicilor aferente, fiind date ca exemple modele ce reflectă particularităţile din Polonia [11.5/A4], Burkov în [11.5/A5], dezvoltînd unele rezultate mai vechi, prezintă o serie de principii noi în conducerea economiei planificate, împreună cu exemplificări, în timp ce Zhao, în [11.5/A1] abordează o aplicaţie a metodei controlului optimal ierarhizat în analiza unui model dinamic de tip input-output.

Sistemele socio-economice sînt sisteme complexe de mari dimensiuni care surprind realităţi ale căror dinamică şi legi de evoluţie nu sînt decît în mică măsură cunoscute, dar problemele legate de conducerea acestor sisteme sînt vitale în contextul abordării unei planificări la nivel naţional. Apare astfel ca necesară o evaluare a potenţialului metodologic şi instrumentar, în vederea stabilirii unor politici convenabile în activităţile sociale şi economice.

Se remarcă o ineficienţă a metodologiilor existente în prezent pentru studiul sistemelor socio-economice, la care se adaugă aportul insuficient al instrumentelor matematice disponibile. Lipsa unor exprimări cantitative — sub forma unor ecuaţii matematice — asupra dependenţelor şi a dinamicilor disciplinelor specifice a condus la o alternativă mixtă, de combinare a descrierilor cantitative şi calitative în cadrul sistemelor de tip om-maşină. Aceste noi structuri decizionale au în vedere utilizarea modelelor cu structură variabilă, crearea rapidă de modele de mici dimensiuni, dar conforme unor reguli predefinite de clasificare, preconizîndu-se de asemenea construirea de modele specifice fiecărui nivel ierarhic, adică modele interconectate ce agregă şi dezagregă procesul decizional, într-un cuvînt se încearcă realizarea unei metodologii adecvate de modelare.

Orice ştiinţă tinde să încorporeze cît mai multă matematică şi să-şi crească numărul legilor specifice exprimate sub forma unor descrieri matematice. Privind retrospectiv la ştiinţele naturii, se remarcă un progres al metodelor cantitative (exprimate prin ecuaţii matematice). Se pune însă întrebarea dacă activităţile socio-economice pot fi suficient de bine explicate şi analizate cu metode cantitative. Răspunsul dat de practică este negativ, impunîndu-se în acest sens noi metode, pe care le vom numi *calitative*. Pentru descrierile calitative se preconizează utilizarea tehnicilor „fuzzy" care permit în final o abordare matematică a informaţiilor imprecise, vagi. Astfel poate fi măsurat gradul de posibilitate precum şi gradul de satisfacere în cadrul unor discipline precum sociologia, economia, psihologia, politica etc., deci acolo unde se constată imposibilitatea unor exprimări cantitative. Se revine astfel la conceptul de satisfacere, dezvoltat de Simon în [1] ca o alternativă a conceptului de optimalitate.

Referindu-se la rezolvarea problemelor socio-economice, Liu semnalează în [11.5/A2] extinderea sistemelor om-maşină şi este propusă o combinare a metodelor cantitative cu cele calitative. În mod cert s-a plecat de la observaţia că pe plan mondial a avut loc o explozie a sistemelor de asistare a deciziei, apărute în ultimii 10 ani, şi care reprezintă soluţia cea mai avansată în conducerea sistemelor mari şi complexe. În cadrul sistemelor suport pentru decizie, modelelor li se acordă un loc important, dezvoltîndu-se tehnici specifice pentru construirea şi rezolvarea lor.

Este de remarcat faptul că în [11.5/A2] sînt argumentate motivele pentru care cercetătorii chinezi, bazîndu-se pe realităţile concrete din ţara lor — perioadă scurtă de tranziţie la economia de tip socialist, diferenţiată istoric în patru etape, suprafaţă geografică mare şi eterogen dezvoltată etc. — au structurat o nouă metodologie de inginerie a sistemelor socio-economice, bazată pe modele de clasificare şi modele cu structură variabilă, punîndu-se accentul pe coordonarea om-maşină în contextul sistemelor de asistare a deciziilor. Sînt semnalate aplicaţii în politica demografică a Chinei şi în agricultură, iar actualmente, pe baza acestei noi metodologii, are loc fundamentarea planului naţional de învăţămînt (educaţie).

R. Kulikowski, reprezentant de seamă al şcolii poloneze de teoria sistemelor şi modelare, prezintă în [11.4/A4] o încercare de modelare a sistemului socio-economic, care este privit ca un integrator al subsistemelor demografic, gospodăresc, forţă de muncă şi economic. Plecînd de la observaţia că majoritatea modelelor utilizate în planificare sînt concentrate pe procese şi politici economice, substratul social fiind eventual reprezentat prin tehnici descriptive sau speculaţii bazate mai mult pe intuiţie decît pe o analiză strictă, Kulikowski dezvoltă o paletă de modele ale sistemului socio-economic ce pot caracteriza nivelul de retribuire, nivelul de educaţie, nivelul asistenţei sociale etc., deci standardul de viaţă al unei populaţii. Lucrarea porneşte de la premisa că politica socială este responsabilă de distribuţia venitului naţional între diferitele grupuri sociale, precum şi la nivelul fiecărei familii. Orice familie îşi adaptează structura, ca răspuns la politica socială, în aşa fel încît resursele de muncă să se dividă între munca exterioară gospodăriei, munca din gospodărie, educarea copiilor etc. Deoarece venitul naţional depinde la rîndul său de nivelul forţei de muncă, rezultă că procesele socio-economice sînt interdependente, factorul dominant fiind politica socială, care poate fi direcţionată în vederea maximizării potenţialului naţional şi personal.

O primă parte a lucrării [11.5/A4] se ocupă de familie, gospodărie şi participarea la muncă a membrilor familiei. Membrii familiei au fost împărţiţi în susţinători şi persoane dependente — adică copii pînă la 24 de ani, fiind prezentate, conform statisticilor, curbe reprezentînd relaţia dintre : x — participarea la muncă a părinţilor, c — raportul dintre numărul de copii şi numărul total al membrilor familiei, şi a — vîrsta mamei. Scopul acestor statistici este de a construi un model care să surpindă relaţia dintre x şi c, precum şi impactul politicii sociale, într-un mod formalizat. Este introdusă *funcţia utilitate*, care caracterizează etalonul familiei medii, influenţînd I — venitul total al familiei, precum şi V — *valorile familiale* de tipul servicii gospodăreşti, educaţia copiilor etc. Aceste două calităţi I şi V sînt exprimate sub forma unor ecuaţii lineare, iar pentru o funcţie utilitate U (I, V) oarecare, care verifică restricţii de concavitate şi diferenţiabilitate natural verificabile, sînt date condiţiile de optimalitate cu ajutorul cărora pot fi determinate cele mai bune strategii pentru a fi atinse valorile optime ale lui x şi c.

În ceea ce priveşte modelul forţă de muncă, se pleacă de la cererea şi oferta existentă. Scopul său este de a ajuta la alegerea celei mai bune politici de distribuire a veniturilor între

diferitele sectoare ale economiei. Sînt exprimate diferenţiat cantitatea de forţă de muncă în mediul rural şi în mediul urban, ţinîndu-se cont de migraţia sat-oraş şi de rata transformării comunelor în oraşe. Aceşti doi factori au fost la rîndul lor exprimaţi prin modele lineare, iar coeficienţii au fost determinaţi prin regresie lineară. În urma efectuării de experimente se observă că, în general, nu poate fi obţinută o stare de echilibru între forţa de lucru rurală şi cea urbană, datorită faptului că oferta este mai mică decît cererea în ambele sectoare. Pentru economia modelată cu ajutorul unei funcţii de producţie de tipul Cobb-Douglas, există posibilitatea determinării cererii optime de forţă de muncă urbană.

Kulikowski expune în final o altă clasă de modele ce se referă la politicile de distribuire a veniturilor. Sînt considerate mai multe criterii : împărţirea veniturilor între generaţii (de exemplu, mărirea alocaţiilor pentru copii faţă de mărirea pensiilor), între mediul urban şi cel rural, iar în cadrul aceluiaşi mediu, între diferitele sectoare. Sînt luate în considerare migrarea sat-oraş şi urbanizarea comunelor, fiind discutate urmările acestor transformări în dinamica costului vieţii. Cîteva strategii de stabilire a raportului retribuţiilor dintre mediul urban şi cel rural sînt discutate. Modelele sînt privite ca instrumente pentru definirea unor politici de dezvoltare a mediului social în conexiune cu cel economic. Aceste instrumente înglobate în sisteme interactive de asistare a deciziei permit trecerea de la adoptarea de hotărîri unilaterale la negocieri între puncte de vedere diferite, cu scopul găsirii unor soluţii unanim satisfăcătoare.

Trebuie remarcat că articolul surprinde foarte clar şi coerent aspecte socio-economice de importanţă vitală în dezvoltarea unei ţări, propunînd tehnici de abordare susţinute cu exemple din statisticile poloneze. Alături de experienţa bogată existentă la noi în problematica socio-economică din cadrul Sistemului Informatic Naţional şi al Sistemului Informatic Teritorial, articolul aduce contribuţii personale în modelarea sistemului social, idei şi rezultate care merită să fie valorificate în vederea determinării unor politici optimale la nivel naţional.

Modelele cu specific economic sînt prezentate în cadrul acestei sesiuni în [11.5/A1], unde este dezvoltat un model dinamic intrare-ieşire de tip Leontief, caracterizat prin ecuaţii diferenţiale prospective, precum şi algoritmul de rezolvare. Tehnica prezentată se constituie într-o metodă generală de planificare optimală a sistemelor dinamice mari cu retardări. Prin alegerea unui criteriu de performanţă pătratic, se obţine o problemă de conducere optimală, iar printr-o tehnică specifică de descompunere se ajunge la un şir de probleme de mici dimensiuni, a căror soluţie rezultă iterativ. Problema economică este clasică, şi după un număr de reformulări se prezintă astfel :

$$\text{Min } J = \sum_{k=0}^{K-1} \left\{ \frac{1}{2} \| X(k) \|^2_{Q(k)} + \frac{1}{2} \| Y(k) - Y_0(k) \|^2_{R(k)} \right\} \qquad (1)$$

cu restricţiile

$$G_k X(k) = \sum_{j=1}^{T} \{ H_{j,k} X(k+j) + Y(k) \} \qquad k = 0, 1, \ldots, K-1 \qquad (2)$$

$$X_{min}(k) \leqslant X(k) \leqslant X_{max}(k) \qquad k = 0, 1, \ldots, K-1 \qquad (3)$$

$$Y_{min}(k) \leqslant Y(k) \leqslant Y_{max}(k) \qquad k = 0, 1, \ldots, K-1 \qquad (4)$$

$$X(0) = X_0 \qquad (5)$$

unde :

$k=0$ este anul iniţial al planului ;
$k=K-1$ — anul final al planului ;
$R(k)$, $Q(k)$ — matrice diagonale de ponderare, pozitiv definite ;
$X(k)$ — ieşirile ;
$Y(k)$ — intrările ;
$Y_0(k)$ — intrări precalculate ;
G_k, $H_{j\,k}$ — matrici care conţin coeficienţi de consum şi investiţii ;
K — orizontul dinamicii ;
T — întîrzierea maximă.

Algoritmul propus pentru rezolvarea problemei (1)—(5) apelează la funcţia Lagrange, care printr-o rescriere convenabilă poate fi descompusă în K subprobleme independente. Deci determinarea soluţiilor X, Y ale problemei (1)—(5) se reduce la aflarea punctelor şa ale lagrangeanului

$$L=\sum_{k=1}^{K-1}\left(\frac{1}{2}\|X(k)\|^2_{Q(k)}+\frac{1}{2}\|Y(k)-Y_0(k)\|^2_{R(k)}+\right.$$

$$\left.+p^T(k)\left[\sum_{j=1}^{T}H_{j,k}X(k+j)+Y(k)-G_kX(k)\right]\right) \tag{6}$$

unde p (k) sînt multiplicatorii lui Lagrange.

Aflarea punctelor şa ale funcţiei L revine la rezolvarea problemei

$$\max_{p}\ \min_{x,\,y} L(x,y,p) \tag{7}$$

Zhao propune o metodă iterativă în doi paşi asemănătoare celei construite de Propoi şi Yadjkin în [2]. Este de semnalat totuşi că în algoritmul prezentat în [11.5/A1] nu sînt specificate condiţii de convergenţă şi finitudine, care din punct de vedere practic au o importanţă deosebită.

În [11.5/A5], Burkov propune o serie de rezultate aplicabile în îmbunătăţirea controlului sectorial, bazate pe teoria sistemelor active. Sînt dezvoltate cîteva proprietăţi specifice sistemelor active : optimalitatea principiului controlului perfect, condiţii asupra mecanismelor funcţionale, principiul reglării prin două canale, estimarea performanţei de înţelegere, proprietăţi ce sînt completate cu studii de caz pentru exemplificare. Ideile prezentate în acest articol sînt proprii şcolii sovietice, şi au fost demarate la Institutul de Ştiinţa Conducerii din Moscova încă în urmă cu 10 ani [3]. Este de semnalat faptul că a fost pus la punct un sistem de asistare a evaluărilor cantitative de înţelegere (comprehensive) a performanţei — ACCORD, ce poate fi utilizat atît în fabrici, cît şi în institute de cercetare.

O primă remarcă asupra lucrărilor prezente în secţiunea 11.5/A este aceea că articolele, plecînd de la consideraţii teoretice, converg spre aplicaţii practice ce nu se restrîng la un anumit domeniu, ci vizează întregul sistem socio-economic. În al doilea rînd se observă o încercare de exploatare a noilor tehnici informatice utilizabile în procesul decizional — sistemele suport pentru decizie, punîndu-se accent pe subsistemul modele şi pe interactivitatea om-maşină. Nivelul ştiinţific ridicat al comunicărilor, utilizarea adecvată a exemplelor şi a figurilor lămuritoare, furnizează tot atîtea argumente pentru lecturarea cu interes a acestora.

Modele pentru conducerea sistemelor mari

Cea de a doua secţiune a colocviului dedicat aplicaţiilor tehnicilor specifice sistemelor mari în conducere şi decizie este consacrată, ca şi prima secţiune, sistemelor decizionale ierarhice, şi cuprinde trei lucrări. Faptul că proceselor decizionale li s-au alocat mai multe secţiuni probează importanţa de care se bucură acest subiect în domeniul aplicaţiilor, atît din partea cercetătorilor, cît şi din partea beneficiarilor potenţiali. Sistemele socio-economice, sisteme de mari dimensiuni, complexe, cu dinamici dificil de formalizat şi structură variabilă, sînt caracterizate de legi de guvernare ce nu sînt decît în mică parte descifrate. Deoarece aceste sisteme constituie o componentă vitală a societăţii în care trăim, necesitatea abordării lor atît din punct de vedere cognitiv (descriptiv), cît şi din punct de vedere decizional (conducere) apare ca o condiţie sine qua non. În această secţiune sînt prezentate trei comunicări ce abordează probleme de modelare, analiză şi conducere a sistemelor economice mari, interconectate [11.5/B1], o tratare metodologică a relaţiei produs-mediu, ca o cerinţă a strategiei de cercetare şi dezvoltare [11.5/B2], precum şi o încercare de modelare a comerţului internaţional şi a încărcării vapoarelor, cu aplicaţii la industria navală [11.5/B3]. Se evidenţează diversitatea problemelor dezbătute în aceste lucrări, numitorul lor comun fiind destinaţia socio-economică a metodologiilor, modelelor şi a rezolvărilor propuse.

În opoziţie cu „teoria modernă a reglării", care vizează analiza şi proiectarea sistemelor centralizate, cadru nefavorabil nici din punct de vedere computaţional şi nici ca implementare, teoria sistemelor mari se preocupă de interacţiunile dintre subsisteme interconectate. În acest context, un sistem mare este descompus ierarhic (avîndu-se în vedere structura sa fizică sau matematică), ca o entitate multinivel ale cărei interacţiuni sînt caracterizate prin legături între nivelele superioare şi cele inferioare. În [11.5/B1], Petrovič propune o metodologie de modelare, analiză şi conducere a sistemelor economice interconectate, utilizînd modele bilineare de mari dimensiuni. Sînt utilizate metode ale geometriei diferenţiale pentru studiul analizei structurale a sistemelor interconectate. În final sînt dezvoltate o serie de modele specifice şi este prezentată problema reglării optimale la două nivele.

De obicei în modelele economice comenzile sînt reprezentate prin termeni lineari aditivi incluşi în ecuaţiile sistemelor diferenţiale. În realitate însă, multe comenzi, cum ar fi rata taxei sau rata creşterii stocurilor financiare, acţionează natural într-un mod multiplicativ. În consecinţă, dacă sînt îndeplinite ipoteze convenabile, acest fapt generează modele bilineare. Mai mult, analiza şi reglarea sistemului economic de mari dimensiuni pot fi realizate cu ajutorul modelelor bilineare, dacă interconexiunea dintre subsistemele microeconomice este modelată ca o variabilă de comandă lineară ce primeşte valori într-o mulţime compactă şi este coordonată dinspre nivelul superior către cel inferior. Datorită faptului că bilinearitatea se prezervă prin dezagregarea sistemelor interconectate, se pot obţine modele particulare bilineare de orice ordin şi nivel de detaliere.

Vom prezenta în continuare modelul bilinear conform [11.5/B1]. Să presupunem că dinamicile subsistemelor sînt reprezentate de ecuaţii diferenţiale lineare :

$$\dot{x}_i(t)=A_i x_i(t)+B_i^1 u_i^1(t) \qquad i=1,\ldots,N \tag{8}$$

$$u_i^1(t)=-K_i^1 x_i(t) \qquad i=1,\ldots,N \tag{9}$$

unde :

$x_i(t)\in R^{n_i}$ — vectorul de stare pentru al i-lea subsistem S_i ;
$u_i^1(t)\in U$ — comenzile de feedback local.

Structura interconexiunilor este determinată de matricele A_{ij} şi vectorii $v_{ij}(t)$, $(i, j=1,\ldots,N)$ Fie $u^c(t)$ comanda dinspre nivelul coordonator către nivelul inferior. În acest context modelul se poate scrie :

$$\dot{x}(t)=Ax(t)+\sum_{\substack{i,j=1\\ i\neq j}}^{N} v_{ij}(t)A_{ij}x(t)+Bu_c(t) \tag{10}$$

unde

$$A=\mathrm{diag}\,(A_1-B_1^1K_1^1,\ldots,A_N-B_N^1K_N^1;$$

şi $x\in R^{N_s}$, $N_s=n_1+n_2+\ldots+n_N$ — vectorul de stare al întregului sistem.

Modelul de mai sus este bilinear, în sensul că este linear în stări şi linear în comenzile $u^c(t)$, $v_{ij}(t)$, dar nu şi împreună. Comenzile aditive $u^c(t)$ şi cele multiplicative $v_{ij}(t)$ reprezintă coordonarea globală respectiv interacţiunile dintre subsisteme. Petrovič descrie un model de creştere multisectorial ierarhizat şi un model economic interconectat, făcînd observaţii asupra posibilităţilor de aplicare efectivă a acestor modele.

Ideea de bază în proiectarea reglării este asigurarea stabilităţii asimptotice şi a optimalităţii sistemelor cu ajutorul metodei dublu nivel, în care nivelul superior intervine asupra structurii procesului prin parametrii de interconexiune cu scopul de a permite stabilizarea şi compensarea perturbaţiilor prin feedback la nivelul inferior. În [11.5/B1] se face şi o analiză structurală a sistemului interconectat utilizînd algebra Lie şi metode ale geometriei diferenţiale. Sînt descrise mulţimile accesibile şi cele structural accesibile pentru sisteme ce pleacă dintr-o stare x la momentul t, Petrovič furnizînd o teoremă care caracterizează mulţimea structural accesibilă a sistemelor bilineare.

În literatură o atenţie deosebită a fost acordată conducerii descentralizate a sistemelor mari, în special a celor lineare. În anumite condiţii, pentru funcţiile bilineare, este dată forma comenzilor optimale atît pentru variabilele de coordonare, cît şi pentru interacţiuni. Rezultă de aici că teoria sistemelor bilineare poate fi utilizată pentru a furniza o metodologie de abordare a sistemelor socio-economice. Chiar dacă aceste sisteme sînt puternic nelineare, incluzînd comenzi multiplicative şi de tip histerezis, precum şi întîrzieri, procesele fundamentale pot fi analizate cu ajutorul modelelor ierarhice binivel, în care comenzile aditive coordonează subsistemele dinspre nivelul superior, iar cele parametrice (multiplicative) reprezintă interacţiuni între subsistemele nivelului inferior. Desigur, se poate ridica obiecţia că, în timp ce Liu [11.5/A] şi Kulikowski în [11.5/A4] argumentează şi justifică din punct de vedere pragmatic necesitatea utilizării modelor de mici dimensiuni, uşor de construit şi de aplicat, folosirea modelelor mari şi ierarhizate propuse de Petrovič apare greoaie şi dificil de implementat. De asemenea, tehnicile de agregare-dezagregare trebuie să verifice condiţii destul de restrictive pentru a fi consistente [4]. Dar utilizarea sistemelor bilineare pare intersantă şi promiţătoare din cel puţin două puncte de vedere. În primul rînd acestea descriu sisteme aproape neliniare, deci mai aproape de realitate decît sistemele lineare, şi în al doilea rînd este de semnalat structura lor variabilă, nerigidă, fapt ce permite o abordare a modelării economice destul de realistă.

O interesantă pledoarie pentru studierea relaţiei produs-mediu de utilizare, cu aplicaţii în domeniul cercetării şi dezvoltării este făcută de Gerencsér în [11.5/B2]. Ideea de bază este că integrarea unui sistem complex în mediul utilizatorului presupune modificări retrospective în sistem şi/sau în mediu. Noutatea metodologiei propuse constă în participarea activă a mediului la rezolvarea acestor cerinţe. Fiind definită problema de rezolvat, mediul care primeşte un sistem complex trebuie ales şi descris conform acestei probleme. Sistemul extins, care include şi funcţii ale mediului, trebuie analizat funcţional şi divizat în subfuncţii. Acestea trebuie clasificate conform consideraţiilor tehnico-economice şi a posibilităţilor de modularizare : configuraţia de bază a sistemului de implementat, opţiunile sistemului şi cerinţele de dezvoltare a mediului.

În timpul proiectării unui produs este necesar să fie luate în considerare aspectele tehnice precum şi cerinţele de tip marketing. Problema alocării resurselor în proiectele de cercetare-dezvoltare a fost abordată de Winkofski în [5], dar modelul binar dezvoltat de acesta nu lua în considerare relaţia produs finit-mediu de utilizare. Mediile caracteristice diferitelor nivele de utilizare a produselor necesită realizarea unor funcţii specifice. Calea tradiţională de rezolvare este modularizarea produsului. Problema poate fi abordată cu succes numai atunci cînd atît cerinţele tehnice (funcţii realizate de fiecare modul), cît şi cele de tip marketing (selectarea modulelor ce vor fi incluse în produs şi a modulelor opţionale ce sînt create de mediu sau achiziţionate separat) sînt satisfăcute simultan.

O primă remarcă ce se desprinde din [11.5/B2] este aceea că, pentru realizarea aceloraşi funcţii, coeficienţii de inteligenţă ai mediului (EIQ) şi produsului (SIQ) variază invers proporţional. Sînt definite trei categorii de relaţii între un sistem şi mediul în care el funcţionează.

1) Sistemul corespunde unui singur tip de mediu.

2) Sistemul se potriveşte cu mai multe tipuri de mediu. În acest context se pot dezvolta trei strategii :

a) SIQ minim şi, deci, EIQ maxim, ceea ce presupune investiţii mari la nivelul infrastructurii ;

b) SIQ maxim şi EIQ minim ;

c) un compromis, bazat pe considerente economice, între SIQ şi EIQ.

În domeniul informaticii, aceasta este situaţia cel mai des întîlnită, în sensul că se are în vedere atît o creştere a nivelului de inteligenţă a mediului în care se introduc sistemele de calcul, cît şi o achiziţionare de tehnică de calcul modularizată, cu o configuraţie care să corespundă cerinţelor mediului.

3) Sistemul este creat ca un produs de piaţă, deci mediul este necunoscut şi doar estimat.

Gerencsér descrie fazele elaborării strategiei de cercetare, dezvoltare şi producere a unor sisteme de tipul celor din categoria a 3-a. Sînt discutate aceste faze în contextul creării şi producerii unui obiect nou. În final este considerat un exemplu structurat pe metodologia propusă. Chiar dacă articolul are un caracter mai mult teoretic, vizînd punctarea unor principii specifice strategiei de cercetare şi dezvoltare în orice domeniu, utilitatea sa constă în faptul că abordează probleme ce sînt destul de rar analizate, dar care au o importanţă aparte în procesul decizional asociat creării şi implementării noului.

Tot în cadrul aplicaţiilor specifice sistemelor mari, cel de-al treilea articol din cadrul acestei sesiuni se referă la problematica comerţului naval din punctul de vedere al proprietarilor şi al constructorilor de vapoare. Industria navală a cunoscut în ultimii ani o serie de dificultăţi datorate în primul rînd ratelor mici de navlosire (încărcarea vaselor), scăderii valorii navelor şi, deci, a creditelor acordate, costurilor tehnologice mari etc. Pau furnizează în [11.5/B3] un model al tendinţei ratei de navlosire utilizabil în industria constructoare de vapoare şi în comerţul marin. Indicii tendinţei ratei de navlosire sînt determinaţi printr-un proces de „tatonare" bazat pe un model de dezechilibru cu o reaşezare eficientă a capacităţilor de transport disponibile. Scopul acestui model este de a servi la fundamentarea deciziilor pe termen lung în ceea ce priveşte structura şi mărimea flotei, vînzările, cumpărările şi reparaţiile.

Sistemul modelat gestionează o singură linie de vapoare ce operează asupra unei flote mixte formată din tancuri petroliere, vase cu încărcătură vrac („bulk carriers") şi cu încărcătură uscată (dry cargo). Ţinîndu-se cont de tonajul vaselor, rutele pe care acestea pot fi alocate, cererea potenţială de transport, capitalul disponibil şi costurile specifice, Pau structurează un model al ratei de navlosire. Determinarea unei rate de navlosire satisfăcătoare se face pe baza unui algoritm de încercări succesive, fundamentat pe reguli specifice industriei navale şi căruia i se pot adăuga eventual condiţii de optimizare. Este de remarcat faptul că modelul poate fi completat cu restricţiile financiare ale companiei, putînd fi corelat cu modelele de la nivel naţional. În urma coroborării cu o analiză financiar-economică şi cu previziuni asupra comerţului mondial, modelul poate satisface trei tipuri de necesităţi:

— fundamentarea deciziilor asupra investiţiilor pe termen lung (structura flotei, vînzări etc.);

— evaluarea structurii flotilelor şi a rutelor concurenţilor;

— estimarea momentelor critice ale deciziilor strategice majore (finanţe, flotă, împărţirea pieţelor).

Acest model este aplicat de către Battelle Memorial Institute, iar noutatea şi ineditul său justifică lipsa bibliografiei.

În contextul în care congresul şi-a propus să propulseze diferite aplicaţii ale automaticii (A Bridge Between Control Science And Technology), lucrările cuprinse în secţiunea 11.5 B se subsumează acestui deziderat, iar interesul pentru subiectele tratate este cu atît mai mare cu cît domeniile abordate corespund unor cerinţe reale, concrete furnizate de practica decizională

Sisteme suport pentru decizie

Presiunea continuă pentru îmbunătăţirea eficienţei proceselor decizionale i-a condus pe decidenţi către tehnologiile informatice de vîrf. În general deciziile adoptate la un anumit moment sînt supuse la două tipuri de intrări. Pe de o parte judecata umană, concretizată prin experienţă, pricepere, bun simţ, şi pe de altă parte ceea ce numim informaţie, adică un set de date la care trebuie să apelăm pentru a lua o decizie. A ne baza doar pe judecata noastră este evident un mod depăşit de luare a deciziilor. A lua decizii bazîndu-ne numai pe informaţii, ceea ce revine la a apela la sisteme expert care hotărăsc independent de voinţa umană, este posibil, cel puţin la nivelul actual al informaticii în lume, numai pe domenii foarte restrictive din realitate. Apare ca un corolar al celor afirmate mai sus faptul că este necesară existenţa unei legături a celor două tipuri de intrări. Experienţa a arătat că o clasă largă de probleme existente în conducerea diverselor sisteme nu se pretează unui suport informatic tradiţional; această arie de probleme, implicînd luarea deciziilor la orice nivel, va putea beneficia de o recentă inovaţie în aplicarea ultimelor dezvoltări ale informaticii, şi anume Sistemele Suport pentru Decizie (SSD).

Secţiunea a treia a colocviului dedicat aplicaţiilor tehnicilor specifice sistemelor mari în procesul decizional şi de conducere este consacrată sistemelor suport pentru decizie. SSD sînt sisteme de procesare a informaţiilor care asistă, adesea interactiv, procesul decizional, constituind sinapsa dintre informaţii, organizate sub forma unor baze de date, şi judecata decidentului. Accentuăm că diferenţa dintre SSD şi sistemele informatice operaţionale constă tocmai în legătura explicită dintre SSD şi procesul de decizie. Rolul acestor sisteme nu este de a înlocui decidentul ci de a-i spori eficacitatea, asistarea acestuia avînd o dublă motivaţie:

— SSD facilitează interacţiunea dintre date, proceduri de analiză a datelor, modele de decizie şi utilizatorii acestora, recurgînd la tehnologii relevante de comunicare, limbaje cît mai apropiate utilizatorului pe domeniul de aplicaţie, grafică interactivă etc.;

— SSD asistă şi decizii nestructurate, pentru care este dificil de furnizat date şi proceduri relevante.

Secţiunea 11.5/C cuprinde un număr de cinci lucrări. Textul celei de a şasea comunicări [11.5/C2] nu este prezent în volum, dar pentru cei interesaţi menţionăm că autorii au dezvoltat acelaşi subiect în două lucrări recente, [6] şi [7]. Articolele sesiunii de faţă, avînd fiecare dezvoltări pe aplicaţii diferite, converg către un numitor comun şi anume demonstrarea faptului că, în conformitate cu experienţa ultimilor 10 ani, cea mai viabilă soluţie în exploatarea tehnologiei actuale a calculatoarelor, avînd scopul de a veni efectiv în întîmpinarea procesului decizional, îl constituie SSD. Singh prezintă în [11.5/C1] cercetări asupra SSD utilizate în procesul de marketing precum şi în conducerea strategică a corporaţiilor multinaţionale. Berezovskiy se ocupă în [11.5/C3] de etapa de concepţie în proiectarea sistemelor tehnice complexe, iar Waśniowski în [11.5/C4] dezvoltă un sistem expert pentru viitorologie, evaluarea impactului şi planificarea strategică cu aplicaţii în domeniul cercetării viitorului din punct de vedere socio-economic, tehnologic etc. În [11.5/C5], Aven descrie un SSD utilizabil în transporturile marinei comerciale şi implementat la Baltic Shipping Agency, iar Baba abordează în [11.5/C6] trăsăturile structurale şi comportamentale a două tipuri de corporaţii, „homeostatice“ respectiv „dinamice“, cu scopul de a perfecţiona strategiile de introducere a noului în corporaţiile bazate pe tehnologii.

Pentru a caracteriza mai bine sistemele de asistare a deciziei trebuie menţionat faptul că ele reprezintă o fuziune între sistemele informatice, tehnicile de inteligenţă artificiale, cercetările operaţionale şi ştiinţa conducerii, [8]. În realizarea unui SSD, punîndu-se accentul pe descrierea procesului decizional, baza sa teoretică va fi atît informatică, cît şi de cunoaştere psihologică, acesta avînd ca menire să asiste intuiţia, să includă atît valorile, cît şi logica realităţii, să facă faţă ambiguităţii, lipsei de informaţii, multiplicităţii şi multidimensionalităţii. În acest context se situează şi punctul de vedere al lui Singh, care în [11.5/C1] afirmă că SSD constituie inima noii generaţii de sisteme de birotică. Într-o primă parte a lucrării, Singh dezvoltă un model al legăturii producător-vînzător cu amănuntul - cumpărător, cu scopul optimizării lanţului de marketing. Facem observaţia că modelul considerat este static, în sensul că se bazează pe următoarele ipoteze : toată producţia realizată într-o perioadă se contractează cu vînzătorii, care la rîndul lor vînd totul consumatorilor. Deci în acest lanţ nu există nici un fel de stocuri. Pe acest model sînt prezentate studii de simulare, făcîndu-se analiza rezultatelor obţinute. Programul de suport al deciziei include facilităţi de schimbare interactivă a parametrilor, ceea ce permite decidentului, beneficiind de asemenea şi de facilităţi grafice, să aleagă varianta cea mai convenabilă. Programul este implementat în FORTRAN 77 pe un minicalculator Prime 750. În a doua jumătate a comunicării, Singh atacă problematica corporaţiilor multinaţionale. Acestea sînt privite ca nişte sisteme ierarhizate, pe nivelul superior aflîndu-se conducerea corporaţiei, iar la nivelul inferior managerii diferitelor filiale, care sînt supuşi la diverse presiuni atît din partea conducerii corporaţiei, cît şi din partea ţărilor în care operează, presiuni care în general se manifestă antagonist şi care au o puternică dinamică în timp. Pentru astfel de probleme, cu o structură complexă, există în literatură o clasă de modele conceptuale lingvistice, care descriu verbal sistemul şi pentru care au fost create SSD specifice, [9]. Singh încearcă însă o abordare cu ajutorul modelelor matematice dinamice de optimizare cu restricţii lineare şi funcţii obiectiv pătratice. Pentru rezolvarea acestora se recurge la funcţia lui Langrange asociată problemei iniţiale şi la teorema de punct şa. În procesul de determinare a punctelor şa ale langrangeanului se observă posibilitatea spargerii problemelor de maximizare ce apar în operaţia min-max, într-un număr de subprobleme independente egal cu numărul filialelor corporaţiei. Este prezentată în final o argumentare detaliată a modelului. Desigur, cele două modele descrise au o importanţă deosebită prin ele însele, dar utilitatea în contextul decizional se amplifică o dată cu încorporarea lor în instrumentele de suport al deciziei. Să observăm că Singh propune doar un cadru pentru SSD, care, prin menirea sa, trebuie să conţină o paletă diversă de modele ce asistă utilizatorul în luarea deciziilor, şi nu numai un singur tip de modele. Dar, în acelaşi timp, să, menţionăm că dacă SSD se împart în sisteme orientate spre date şi sisteme orientate spre modele, prima categorie furnizînd facilităţi de regăsire, analiză şi prezentare a datelor, în timp ce a doua, mai sofisticată, oferă în special modele de optimizare şi simulare, Singh preconizează un SSD care să aparţină celei de a doua categorii.

Berezovskiy, abordînd în [11.5/C3] domeniul proiectării asistate de calculator, se ocupă de faza de concepţie şi alegere a unei alternative satisfăcătoare, în contextul în care sînt numeroase criterii ce trebuie îndeplinite, dintre care multe dificil de formulizat. Este dezvoltată o

metodă de determinare a preferinţelor în vederea alegerii unei opţiuni „nedominate", adică pentru care nu există altă opţiune superioară din toate punctele de vedere. Sînt prezentate rezultate teoretice care se concretizează în două teoreme ce evidenţiază tehnici specifice de descompunere în analiza opţiunilor.

Am arătat anterior că unul din obiectivele primordiale ale SSD îl constituie aplicarea intensivă a tehnicilor de inteligenţă artificială. În acest context Waśniowski propune în [11.5/C4] un sistem expert bazat pe cunoştinţe destinat cercetării viitorului, evaluării impactului şi planificării strategice. Sistemul expert se compune din două module : baza de cunoştiinţe şi aparatul inferenţial. Baza de cunoştinţe conţine informaţiile despre domeniul unei problematici particulare, iar aparatul inferenţial rezolvă problemele formulate de utilizator folosind baza de cunoştinţe şi generează explicaţii asupra soluţiilor adoptate. Deoarece în viitorologie intervin numeroase evenimente, tendinţe şi multe alte fapte dificil de identificat şi de corelat, a fost dezvoltat un sistem denumit „FUTURIST" cu ajutorul căruia se pot construi modele calitative de simulare. Cunoştinţele pentru rezolvarea problemelor sînt reprezentate în termenii unor probabilităţi condiţionate, adică de tipul : anumite evenimente vor avea loc dacă în prealabil alte evenimente au avut loc. În acest fel pot fi create scenarii utilizînd diferite modele sau experienţa umană, într-un mod interactiv. Programul, scris în PROLOG, a fost implementat pe un minicalculator PDP-11 sub un sistem de operare de tipul RSX-11M. Este de menţionat faptul că sistemul utilizează raţionamente cu predicate fuzzy, răspunde la întrebări de tipul „Why ?" şi are o comandă „HELP", deci poate fi considerat ca fiind un sistem orientat către utilizator. Trebuie să menţionăm că şi la noi în ţară există experienţe asemănătoare [10], [11], în sensul că au fost create cîteva tipuri de sisteme experte dintre care unele, prin posibilitatea de a răspunde şi la întrebări de tipul „How to ?" şi „What if ?", au facilităţi superioare celui descris de Waśniowski.

Dacă în [11.5/B3] Pau a prezentat doar un model al ratelor de navlosire (încărcarea vaselor) cu aplicaţii la industria navală, Aven descrie în [11.5/C5] un instrument complex pentru luarea deciziilor în conducerea transportului marin comercial. Transportul marin este analizat în conexiune cu celelalte sectoare ale economiei naţionale, avînd atît trăsături comune cu acestea, cît şi caracterisrtici spec'fice. Faptul că procesul este continuu, are un grad mare de variabilitate iar factorii exogeni sînt impredictibili, precum şi natura sa probabilistă obligînd la actualizări continue impun cu necesitate utlizarea unui SSD. Sistemul este destinat ordonanţării traficului naval, a reparaţiilor, a operaţiilor în uzinele de reparaţii etc. şi este gîndit ca o interconectare de mai multe subsisteme. Două dintre acestea, sistemul de ordonanţare continuă şi sistemul informatic pentru containere navale sînt prezentate în detaliu. Sistemul de ordonanţare continuă este dedicat conducerii traficului naval pe o durată pînă la 120 de zile. Actualizările se fac pe baza unei reţele de videoterminale, datele fiind gestionate de un SGBD de tipul INES-2M. Etapele dialogului dintre om şi sistem constau în identificarea utilizatorului, specificarea scenarului cadru şi efectuarea de simulări cu diferite scenarii utilizînd programe specifice de aplicaţii. Sistemul de ordonanţare cuprinde 15 baze de date cu aceeaşi structură logică. Sistemul „Container" este structurat pe trei nivele : containere, agenţia navală şi minister. Sînt luate în considerare toate operaţiile pornind de la transportul pe cale ferată, încărcarea, descărcarea, distribuirea sosirilor containerelor la locurile de încărcare etc. Să semnalăm faptul că sistemul include pachete de programe pentru rezolvarea problemelor de optimizare, alături de facilităţile amintite pentru simulare. Acest produs informatic, care este implementat la Baltic Shipping Agency, este — atît prin facilităţile pe care le pune la îndemîna utilizatorului, cît şi prin complexitatea şi dificultatea problematicii pe care o asistă în procesul decizional — unul din cele mai reprezentative sisteme de suport al deciziei create pe plan mondial. La noi în ţară există încercări de a dezvolta un astfel de sistem cu facilităţi deosebite [12], sistem care include în plus şi tehnici de inteligenţă artificială precum şi elemente de ştiinţa conducerii.

Baba, în [11.5/C6] prezintă relaţiile care există între două categorii de corporaţii : cele „homeostatice", care caută să-şi menţină structura invariabilă prin controlul ce-l impun asnpra pieţii, şi cele „dinamice", care încearcă să exploateze şi să constituie avangarda efervescenţei tehnologice. Schimbările tehnologice sînt privite ca forţe potenţiale generatoare de perturbaţii în mediul corporaţiilor şi în acest context sînt avansate ipoteze asupra avantajelor corporaţiilor tinere faţă de corporaţiile mature. Analogia cu evoluţia umană este folosită pentru a explica adaptările

corporațiilor la tehnologiile noi, fapt ce presupune riscuri mari. Acest articol, cu toate că vizează luarea deciziilor cu scopul îmbunătățirii structurii corporațiilor, ca și a politicilor și a strategiilor acestora, nu își propune să abordeze instrumentele practice necesare desfășurării procesului decizional, el încercînd doar, pe baza unei documentații serioase, să prezinte o serie de principii și observații.

Dintre lucrările congresului, mai pot fi amintite și alte comunicări ce au avut ca obiectiv prezentarea unor sisteme de suport a deciziei. Astfel poate fi menționată comunicarea lui Amano [11.6/A4], în care este propus un sistem de planificare asistat de calculator, utilizabil în rețeaua de transport public urban, cu ajutorul căruia se poate predicționa fluxul traficului în rețea. Sistemul este utilizat în evaluarea impactului produs într-un mediu urban de construirea unui metrou. În [11.9/2] Sheridan pledează în favoarea SSD, subliniind rolul cercetării de laborator în domeniul sistemelor de reglare de tip om-mașină. În domeniul exploatării apelor sînt de semnalat o serie de sisteme ce sînt utilizate pentru tratarea apelor reziduale, Maeda în [11.7/5], sau pentru evaluarea lingvistică a resurselor de apă, Bečvár în [11.8/A3], sistem ce este bazat pe teoria mulțimilor fuzzy. În sfîrșit, Carmon propune în [11.8/C2] un sistem interactiv care realizează modele pentru diferite proiecte, un model de autoînvățare al sistemului, ce utilizează informația din timpul procesului decizional, un model de examinare a diferitelor politici, un set de procedeuri în ajutorul utilizatorului (regăsirea informației relevante din proiecte, atenționări, actualizări, etc.), un algoritm de dialogare cu decidentul.

Chiar dacă primele SSD au fost construite cu numai 10 ani în urmă, succesul acestora a fost cel scontat, fapt demonstrat de numărul din ce în ce mai mare de astfel de sisteme utilizate astăzi. Să precizăm că SSD sînt folosite în procesul decizional al unor domenii foarte diverse și asistă o paletă largă de decizii. Cele mai numeroase sisteme sînt exploatate în probleme de planificare la nivel de corporații, sau la nivel guvernamental, precum și în domeniul militar, în medicină, în conducerea băncilor etc. La noi în țară există realizări în această direcție de vîrf a informaticii, dintre care menționăm sistemul „DISPECER", utilizat cu succes în conducerea proceselor tehnologice [13], precum și sistemul „CAMDS", un SSD bazat pe modele, dedicat în special conducerii sistemelor socio-economice [12].

BIBLIOGRAFIE

1. *Simon, H.A.* **The Sciences of the Artificial,** MIT Press, Cambridge, Massachussetts, 1969.
2. *Propoi, A.I. Yadjkin, A.B.* **Metodă iterativă parametrică pentru soluționarea problemelor de programare liniar-dinamice** (lb. rusă), Avtomatika i Telemehanika, Nr. 2, 1976.
3. *Burkov, V.N. Kondratuev, V.V.* **Quasioptimality of openhanded control in resource allocation,** Preprints "IFAC 6-th Congress", Pittsburg, 1975
4. *Sondermann, D.* **Optimale Aggregation von grossen Gleichungen Systemen,** Zeitschrift für Nationalökonomie, 33, 1973.
5. *Winkofski, E.P. Baker, N.R. Sweeney, D.J.* **A Decision Process Model of R & D Resource Allocation in Hierarchical Organizations,** Management Science, Vol. 27, No. 3, 1981.
6. *White, C.C., Sage, A.P., Scherer, W.T.* **Decision Support with Partially Identified Parameters,** Large Scale Systems, Vol. 3, 1982.
7. *Lagomasino, A., Sage, A.P.* **A Learning Approach for Incorporation of Imperfect Knowledge in Decision Support. System Design,** First European Workshop on the "Real Time Control of Large Scale Systems", Patras, July 1984.
8. *Gheorghiu, O.* **Sistemele Suport Pentru Decizie – Instrumente Sofisticate În Ajutorul Procesului Decizional,** Sesiunea Națională de Comunicări Științifice, ICI-București, 1984.
9. *Negoiță C.V.* **Fuzzy Sets in Decision Support Systems,** Working Paper, 1982.
10. *Bărbuceanu, M.* **An Object-centered Framework for Expert Systems in CAD,** Proceedings of the IFIP WG. 5.2 Budapest, 1984
11. *Georgescu, I., Hotăran, A., Predescu, R., Vlasiu, M., Petrescu, F., Nicolăiță D., Nagy, A.* **INTEXP – A Domain Independent Expert System,** Raport de cercetare, ICI, 1984.
12. *Gheorghiu, O.* CAMDS—A Model Basel DSS, Raport de cercetare, ICI, 1984.
13. *Filip, F.G., Donciulescu, D.A., Orășanu, L.* **Industrial Applications of Hierarchical Optimization** Methods, in A. Straszak (ed), **Preprints,** 3-rd IFAC/IFORS Symp. Large Scale Systems Theory and Applications, Warsaw, 1983.

METODE DE DECIZIE MULTICRITERIALE

Portofoliul de metode multicriteriale este din ce în ce mai vast, acest fapt fiind determinat de cerinţele practice din ce în ce mai complexe ale modelelor reale.

Metodele multicriteriale se clasifică în: metode de decizie multiobiectiv (MODM) şi metode de decizie multiatribut (MADM). În metodele de decizie multiobiectiv, obiectivele sînt explicite, iar alternativele sînt definite implicit prin restricţii, şi pot fi oricît de multe. În metodele de decizie multiatribut, obiectivele sînt implicite şi greşit definite, alternativele sînt explicite şi puţine la număr.

Dintre metodele MODM, cele mai performante şi mai des utilizate sînt metodele interactive. Aceste metode implică intervenţia decidentului prin dialogul lui cu maşina/analistul la fiecare iteraţie. Aceasta conferă algoritmilor o mai mare flexibilitate şi o mai corectă oglindire a procesului de decizie.

Din punct de vedere matematic, putem asimila decidentul cu un operator fuzzy, ceea ce, în anumite condiţii, fie de situaţie "limită", „critică", fie din motive algoritmice, poate determina noncenvergenţa metodei. Pentru a evita aceasta, cît şi alte inconveniente rezultate din aplicarea metodelor interactive, se prevede fixarea de către analist, prin programul software, a unui "domeniu de desfăşurare", în cadrul căruia procesul este convergent şi rezonabil.

Prezentăm cele cinci articole din secţiunea *11.5/D* "Metode de decizie multicriteriale" prezentate la simpozionul IFAC 1984. Se remarcă o preocupare constantă în elaborarea unor, metode interactive (cu un decident şi/sau cu mai mulţi decidenţi) care să prezinte caracteristici de performanţă superioare altor metode. De asemenea, în [11.5/D4 şi 5] se face o analiză, vizînd globalul, a problemelor incertitudinii şi a metodelor de decizie multicriteriale cu mai mulţi decidenţi.

În acest paragraf, încercăm să prezentăm pe scurt şi să facem o analiză comparativă a metodelor prezentate de Nakayama şi Sawaragi, Gomide, Tarvainen şi Haimes, Stanoulov, Aleskerov şi Vol'skiy, Takeguchi şi Akashi la simpozionul IFAC 1984.

Astfel, Nakayama şi Sawaragi prezintă în [11.5/D1] o metodă de decizie multiobiectiv iterativă, de tipul metodelor din familia STEM [1]. Metoda prezentată în [11.5/D1] este mai flexibilă şi mai rapid convergentă decît metodele similare. Aceasta se realizează prin faptul că decidentul are posibilitatea să crească şi/sau să relaxeze un număr oarecare de obiective, procedeul putîndu-se repeta, în funcţie de un anumit test, repetarea lui micşorînd numărul de optimizări auxiliare şi asigurînd o convergenţă mai rapidă. A se nota că metoda STEM [1] nu permite decidentului decît relaxarea unui singur obiectiv în cadrul unei iteraţii.

Faţă de metoda STEM, metoda "de satisfacere a compensărilor" a lui Nakayama şi Sawaragi prezintă următoarele facilităţi:

a) mulţimea restricţiei a problemei iniţiale nu este modificată pe parcursul algoritmului;

b) informaţiile cerute decidentului nu necesită un efort prea mare din partea lui, asigurîndu-se totodată un proces de învăţare pentru utilizator;

c) în anumite condiţii se poate obţine o soluţie mai bună decît în metoda STEM, şi anume o soluţie Pareto satisfăcătoare;

d) decidentul poate să relaxeze şi/sau să crească anumite obiective, după cum cere funcţia sa de utilitate implicită, în timp ce metoda STEM permite decidentului doar relaxarea unui singur obiectiv;

e) repetarea eventuală a procedeului de la punctul (d) asigură un număr mai mic de optimizări auxiliare, deci o convergenţă mai rapidă decît metoda STEM.

Faţă de alte metode din familia STEM, cum ar fi SEMOPT [6] care cere rezolvarea a $n \geqslant 2$ optimizări auxiliare la fiecare iteraţie, superioritatea metodei "de satisfacere a compensărilor" este evidentă.

În concluzie, metoda iterativă [11.5/D1] prezintă următoarele caracteristici:

— efort minimal al decidentului;

— informaţia furnizată decidentului este uşor de înţeles;

— convergenţă rapidă;

— număr mic de optimizări auxiliare;

— un proces de învăţare pentru decident.

După cum este arătat în material, această metodă stă la baza unor programe de calcul şi a fost aplicată în proiectarea construcţiei antiseismice a sistemului de piloni pentru un pod lung suspendat.

K. Tarvainen îşi continuă în [11.5/D2] împreună cu F. Gomide şi Y.Y. Haimes, activitatea de cercetare relativ la problemele multicriteriale. De remarcat, dintre lucrările lui, teoria dualităţii pentru problemele convexe multiobiectiv [9].

În [11.5/D2] se prezintă o metodă de negociere iterativă relativ la sistemele mari cu mai multe obiective. Această metodă face parte din clasa MODM cu $n \geqslant 2$ decidenţi.

Autorii clasifică metodele de negociere multicriteriale astfel:

a) metode cu bază matematică (impun satisfacerea anumitor axiome, indiferent dacă sînt sau nu convenabile pentru decidenţi);

b) metoda de generare (se generează toate sau aproape toate soluţiile Pareto, dintre care decidenţii aleg, prin vot, de exemplu, cele mai bune soluţii);

c) metode de decizie spaţială (corespund metodelor cu preferinţe a priori relativ la informaţii şi un singur decident. Au fost concepute iniţial de Robertson, 1976);

d) metode de negociere iterativă.

Metoda prezentată la [11.5/D2] prezintă similitudini cu metoda SEMOPT [6] — un singur decident, şi anume:

1) la fiecare iteraţie se rezolvă un număr de $n \geqslant 2$ probleme auxiliare (dacă N este numărul de decidenţi se rezolvă fie N+1, fie N optimizări auxiliare, în funcţie de varianta folosită).

2) procedeul este convergent doar în cazul satisfacerii anumitor ipoteze.

De remarcat că această metodă este mai flexibilă, respectă opţiunile decidenţilor şi reflectă mai corect procesul real decît metodele din clasele a), b), c), ceea ce îi conferă criterii de performanţă mai ridicate.

De asemenea, prin structura procedeului, poate fi uşor adaptată pentru cazul unui singur decident, ceea ce determină, din punct de vedere al argumentelor matematice, superioritatea acestei metode faţă de SEMOPT (numărul de optimizări auxiliare pentru fiecare iteraţie este mai mic). De altfel, autorii dau un exemplu numeric (riul ipotetic Bow din [5]), care confirmă convergenţa mai rapidă decît cea a metodei SEMOPT.

Metodele MADM (metode de decizie multiatribut) sînt abordate în [11.5/D3] şi [11.5/D4], ultimul avînd o viziune globală şi de sinteză asupra lor.

În [11.5/D4], F.T. Aleskerov şi V.I. Vol'skiy descriu regulile Pareto ca fiind clase de proceduri "de agregare" a criteriilor într-un singur criteriu "de selecţie" rezultant.

Autorii fac o clasificare a acestor proceduri, astfel:

1) *Proceduri de tip I* — care transformă o totalitate de relaţii binare C_i într-o relaţie "de selecţie" C^* rezultantă.

Se definesc operatorii locali care transformă o colecţie de relaţii binare slabe într-o relaţie binară aciclică/tranzitivă/de tranzitivitate negativă.

Se defineşte legătura existentă între acest tip de proceduri şi metodele care folosesc matrici similare cu cele din teoria jocurilor.

2) *Proceduri de tip II* — transformă o totalitate de relaţii binare/criterii într-o funcţie de selecţie rezultantă $C^*(\cdot)$.

Autorii leagă rezultatele obţinute în privinţa acestor proceduri cu cele obţinute la punctul (1), făcînd referinţe la metodele care selectează o submatrice "dihotomică" din matricea totală a alternativelor (vezi metoda prezentată în [11.5/D3]).

3) *Proceduri de tip III* — transformă o totalitate a funcţiilor de selecţie $C_i(\cdot)$ într-o funcţie de selecţie globală $C^*(\cdot)$.

Acest tip de proceduri sînt foarte recente, primele rezultate fiind date de Grether şi Plott în 1982.

N. Stanoulov în [11.5/D3] descrie un procedeu de rezolvare a problemelor MADM pentru cazurile un singur decident şi mai mulţi decidenţi, care aparţine clasei procedurilor de tip II, descrise mai sus [11.5/D4].

Astfel, Stanoulov construieşte un operator local care duce o mulţime de relaţii binare slabe într-o relaţie binară aciclică. Această observaţie plasează metoda în clasa procedurilor de tip I, dar faptul că din matricea alternativelor totale se selectează o submatrice (numită "diho-

tomică") a alternativelor "preferabile", determină apartenența acestei proceduri la clasa celor de tip II, care include strict, într-un anume sens, pe cea a procedurilor de tip I.

Metoda DIMCO (multiple criteria assessment and choice of alternatives via matrix dichotomy [11.5/D3]) poate fi totodată asimilată cu un joc matriceal în care sînt determinate alternativele "cele mai bune", prin selectarea unui șir de "învingători" în cele trei faze ale algoritmului.

Față de alte metode similare (Toda, Amano, 1981) structura de preferință definită între alternative este diferită, ceea ce determină un grad de performanță mai ridicat.

Un studiu asupra incertitudinii deciziilor în condiții de risc este făcut în [11.5/D5], unde T. Takeguchi și H. Akashi presupun că există informații incomplete asupra probabilităților stărilor și, în funcție de tipul acestor informații, definesc dominanța statistică, stohastică sau S-S (statistică și stohastică) a alternativei A1 asupra alternativei A2.

Este, în fond, o tehnică auxiliară prognozei, vizînd o simulare a comportării sistemului real în condiții de incertitudine.

Autorii derivă din inegalitatea Fishburn două inegalități care definesc parțial probabilitățile stărilor :

$$\text{(i)}\ DH_h \geq P_h - P_{h+1} \geq DL_h,\quad h=\overline{1,\ m-1}$$

sau

$$\text{(ii)}\ RH_h \geq \frac{P_h}{P_{h+1}} \geq RL_h,\quad h=1,\ m-1$$

unde P_h este probabilitatea stării h ;

DH_h, DL_h — margini superioare, respectiv inferioare, pentru diferența $(P_h - P_{h+1})$;

RH_h, RL_h — margini superioare: respectiv inferioare, pentru fracția $\left(\frac{P_h}{P_{h+1}}\right)$.

Se definește dominanța statistică a alternativei A1 asupra alternativei A2, ca fiind determinată de inegalitatea :

$$E\,[A1] \geq E\,[A2]$$

unde

$$E\,[A_i] = \sum_{j=1}^{m} P_j U_j^i,\quad i=1,\ 2$$

Autorii derivă din definiția de mai sus și din informațiile incomplete asupra probabilităților stărilor (i) sau (ii) rezultate echivalente pentru a exprima dominanța statistică.

Dominanța stohastică se deosebește de cea anterioară prin faptul că, în locul probabilităților descrise mai sus, folosește distribuții de probabilitate ale consecințelor și introduce de asemenea clase de funcții de utilitate.

Funcțiile de utilitate pot fi clasificate după cum :

1) decidentul este împotriva riscului ;
2) decidentul caută riscul ;
3) decidentul dorește descreșterea riscului ;
4) decidentul este împotriva riscului în "domeniul cîștigului" și pentru risc în "domeniul pierderii".

Distribuția de probalitate poate fi interpretată ca fiind "gradul de risc al alternativelor, din punct de vedere al consecințelor lor".

Spunem că distribuția de probabilitate F a alternativei A1 domină stohastic distribuția de probabilitate G a alternativei A2 dacă

$$E\,(u,\ F) \geq E\,(u,\ G)\quad \forall\ u \in U_j$$

unde

U_j = clasa j a funcțiilor de utilitate

şi

$$E(u, F) = \int_{x \in X} u(x)\, dF(x)$$

$$E(u, G) = \int_{x \in X} u(x)\, dG(x)$$

Cu alte cuvinte, definiţia de mai sus reflectă faptul că alternativa A1 este mai bună decît alternativa A2 din punctul de vedere al "gradului de risc al consecinţelor".

Dominanţa S-S (statistică şi stohastică) poate fi exprimată în următorii termeni : În ce condiţii alternativa A1 este mai bună decît alternativa A2 din punctul de vedere al "gradului de risc al consecinţelor", cînd avem informaţii incomplete asupra probabilităţilor stărilor şi funcţia de utilitate implicită a decidentului aparţine unei clase bine definite matematic ?

Vizînd materialul [11.5/D3], putem aplica acest studiu metodei DIMCO, folosind această tehnică ca o prognoză şi simulare a evoluţiei sistemului global în condiţii de risc.

Rezultatele recente sintetizate aici sînt de două tipuri :

a) de analiză şi sinteză a unor studii anterioare la care se adaugă contribuţii originale [11.5/D4 şi 5]

b) de prezentare a unor metode originale MODM sau MADM [11.5/D1, 2, 3] care extind, în sensul unor performanţe mai ridicate, metode similare deja existente.

Aceste studii devin utile cercetării în acest domeniu prin performanţele ridicate ale metodelor, care le fac mai uşor abordabile computaţional şi mai flexibile în utilizarea lor practică, sintezele, la rîndul lor, asigurînd totodată o deschidere mai largă a metodelor multicriteriale şi o utilizare a lor vizînd globalul.

BIBLIOGRAFIE

1. *Benayoun, R., de Montgolfier, J., Tergny, J., Larichev, O.* — **Linear Programming with Multiplé Objective Functions : Step Method (STEM)**, Math. progr., V.1, No. 3, pp 366—375, 1971.
2. *Blin, J.M.* **A linear assignment formulation of the multiatribute decision problem**, RAIRO, Recherche opérationnelle, V. 10, No. 6, pp. 21—32, 1976.
3. *Delalieux, Y.* **Multiple criteria optimization with adaptive partition**, Preprints of the First European Workshop on the „Real Time Control of Large Scale Systems" (Schmidt, C., Singh, M., Titli, A., Tzafestas, S., ed.), pp. 118—123, University of Patras, Greece, July 1984.
4. *Hwang, C.L., Yoon, K.* **Multiple Attribute Decision Making — Methods and Applications**, Lectures Notes in Economics and Mathematical Systems, Springer — Verlag, 1981.
5. *Hwang, C.L., Masud, A.S.M.* **Multiple Objective Decision Making — Methods and Applications**, Lectures Notes in Economics and Mathematical Systems, Springer — Verlag, 1979.
6. *Monarchi, D.E.* **Interactive Algorithm for Multiple Objective Decision Making**, Technical Reports on Hydrology and Water Resources (No. 6), University of Arizona, 1972.
7. *Siskos, J.* **Application de la methode UTA I à une probléme de selection de points de vente mettant en jeu des critères multiples**, RAIRO, Recherche opérationnelle, 1983.
8. *Steuer, R.E.* **Goal Programming sensitivity analysis using interval penalty weights** — Math. progr., 17, 1979.
9. *Tarvainen, K.* **A preference — oriented duality theory for multiobjective optimization**, Report B70, System Theory Laboratory, University of Technology, Helsinki, 1982.
10. *Vincke, Ph.* **Une méthode interactive en programation linéaire à plusieurs functions économique**, RAIRO, Recherche opérationnelle, V. 10, No. 6, pp. 5—20, 1976.
11. * * * **An International Journal Computers & Operations Research** (J. Raff, ed.), V. 7, No. 1—2, 1980 („Mathematical Programming with Multiple Objectives").

OPTIMIZĂRI ÎN RAMURA TRANSPORTURILOR

Ing. *M. Sîrbu*
I.P.A.

1. INTRODUCERE

Transportul are un rol vital pentru desfăşurarea în bune condiţii a vieţii social-economice dintr-un oraş. În ultimii ani s-a observat că traficul a devenit din ce în ce mai aglomerat, acest lucru datorîndu-se în mare măsură creşterii numărului de maşini particulare. Studiile efectuate au pus în evidenţă faptul că mijloacele „convenţionale" de transport (metrou, autobuze, tramvaie etc.) oferă încă posibilităţi mari de a fi îmbunătăţite.

Lucrările prezentate în cadrul *secţiunii 11.6* tratează cîteva dintre problemele ridicate de aglomerarea circulaţiei. Astfel, sînt propuse metode de descongestionare a traficului urban pe baza utilizării unor sisteme de planificare a reţelelor de transport public, pe baza utilizării unor siteme de dirijare a circulaţiei şi a unor sisteme de comandă corelată a semafoarelor de circulaţie etc.

Unele lucrări propun automatizarea mijloacelor de transport (a maşinilor, a unor mijloace de transport naval), ca o modalitate de îmbunătăţire pe ansamblu a traficului.

2. SISTEME PENTRU TRANSPORTUL TERESTRU

Lucrarea 11.6/A-1 prezintă un sistem ierarhic descentralizat pentru dirijarea circulaţiei rutiere. Prima realizare fizică a sistemului propus, şi care este prezentată pe scurt în lucrare, este sistemul „PROGETTO TORINO".

Un astfel de sistem este necesar pentru satisfacerea următoarelor cerinţe :

a) prioritate absolută pentru liniile de transport public ;
b) îmbunătăţirea semnificativă a mobilităţii vehiculelor personale ;
c) controlul continuu al vehiculelor publice aflate în acţiune ;
d) informarea utilizatorilor care aşteaptă în staţii în legătură cu sosirea următorului vehicul public ;
e) asigurarea unei fiabilităţi ridicate a întregului sistem, aceasta presupunînd autodiagnosticarea şi informarea automată la postul central de dispecer în legătură cu defectele care pot apărea în orice punct al reţelei semafoarelor aflate sub control.

Pentru satisfacerea acestor cerinţe se impun următoarele specificaţii de proiectare ;

— exploatarea avantajelor oferite de coordonarea semafoarelor de circulaţie ;
— acordarea atenţiei cuvenite fenomenelor lente legate de traficul particular ;
— acordarea priorităţii unor linii de transport public pe străzile rezervate, prin intermediul pronosticării prin puncte a mişcării fiecărui vehicul ;
— libertatea controlului semafoarelor la fiecare intersecţie prin intermediul unor operaţii locale în buclă închisă.

Cerinţele de proiectare impuse mai sus conduc la realizarea unui sistem ierarhic distribuit. Modelul dinamic al traficului este descris prin intermediul a trei modele dinamice : modelul macroscopic al traficului privat, care descrie comportarea liniilor de trafic pe întreaga zonă aflată sub control ; modelul microscopic al traficului privat, care furnizează o reprezentare detaliată a traficului privat la fiecare intersecţie şi modelul traficului public, care descrie în detaliu comportarea fiecărui vehicul de pe liniile prioritare.

Strategia de conducere, aflată în strînsă conexiune cu modelele de mai sus, are ca scop minimizarea timpului total pierdut de vehiculele particulare, impunîndu-se în acelaşi timp restricţia ca vehiculele publice de pe liniile cu prioritate să nu fie oprite la intersecţii semaforizate. De asemenea, trebuie să se ţină cont de restricţiile de siguranţă a traficului şi de durata maximă sau minimă a fazelor semafoarelor.

Deoarece apar perturbaţii puternice, restricţiile locale şi globale fiind bazate pe variabile stohastice, va fi necesară o conducere în buclă închisă. Totodată, datorită complexităţii problemei, va fi necesară aplicarea unor metode de decompoziţie.

Sistemul de conducere va fi ierarhizat pe două nivele. La nivelul superior se realizează conducerea globală, bazată pe informaţiile furnizate de modelul global şi se transmit la nivel inferior regulile care trebuie respectate pentru optimizarea performanţelor globale. Sarcinile la nivel inferior sînt realizate de fiecare dintre mecanismele de conducere bazate pe modelele locale corespunzătoare, utilizînd cu maximum de eficienţă informaţia provenită de la nivelul superior.

Sistemul de conducere va avea următoarea structură :

— o reţea de mecanisme de conducere locale, conectate mutual, astfel încît să reproducă topologia conexiunilor dintre intersecţii ;

— o reţea densă a detectoarelor pentru traficul privat şi public, plasate astfel încît să furnizeze o estimaţie de stare ;

— un mecanism de conducere global, conectat la reţeaua mecanismelor de conducere locale.

Implementarea practică a acestui sistem de conducere cu două nivele ierarhice este reprezentată de sistemul „PROGETTO TORINO". Elementele principale care intră în componenţa acestui sistem sînt : detectoare de vehicule particulare, detectoare de vehicule publice, elemente de acţionare a semafoarelor, mecanismul de conducere locală şi mecanismul de conducere globală.

Sistemul „PROGETTO TORINO" asigură verificarea în practică a conceptelor de proiectare descrise în lucrare. Instalarea acestui sistem a fost definitivată conform planului, în anul 1983. Primele rezultate obţinute fiind bune, s-a trecut la aplicarea largă a sistemului în anul 1984.

Problema descongestionării traficului urban este tratată şi în lucrările 11.6/A-2, 11.6/A-3, 11.6/A-4, 11.6/A-5 şi 11.6/A-7

Aspectele de o mai largă generalitate sînt abordate în cadrul lucrărilor *11.6/A-2* şi *11.6/A-4*, care se ocupă cu probleme de modelare şi reglare a proceselor în sistemele de transport şi respectiv cu descrierea unui sistem de planificare asistat de calculator pentru reţelele de transport public.

În *lucrarea 11.6/A-2* a fost dezvoltată o versiune de bază a sistemului de modelare a transportului pe scară largă. Schema generală a procesului de transport este reprezentată cu ajutorul unui set de blocuri de conducere (BC), care definesc sistemul de conducere structurat şi a setului de blocuri funcţionale (BF), care definesc organizarea tehnologiei de transport. Setul componentelor BF este divizat în subseturi neinteractive, în concordanţă cu structura BC. Tranzacţiile dinamice (TD) asigură toate legăturile între BC şi BF. Pe baza acestei scheme de modelare se realizează descrierea caracteristicilor dinamice.

A fost realizat un limbaj de simulare, care susţine sistemul de modelare. Pentru reprezentarea caracteristicilor modelării simulate s-a utilizat o „metodă cadru". Modelarea bazei de date a fost realizată pentru construirea şi reprezentarea parametrilor de stare ai sistemului. Limbajul special de interogare permite realizarea accesului la date şi a unor manipulări cu datele modelării simulate.

Sistemul de modelare şi mijloacele de programare prezentate au fost utilizate pentru predictarea transportului în condiţiile în care interacţionează diferite moduri de transport. Principalul scop al predictării transportului este de a face posibilă reglarea interacţiunii dintre diferitele moduri de transport. Un astfel de model de predicţie a fost construit pentru sistemul de transport pe căi ferate. În acest caz, principalul scop al predicţiei este de a asigura o distribuţie dinamică a fluxului de vehicule ajunse pentru perioade scurte în porturile marine.

În lucrare se tratează şi problema modelării sistemului de conducere. Automatizarea planificării proceselor de conducere pentru sistemele de transport se face pe baza analizei informaţiei de stare a sistemului, analiză care furnizează indicaţii în legătură cu strategiile de conducere, bazate pe model, care pot fi adoptate.

Ca o aplicaţie, s-a construit un model pentru sistemul de transport urban. Pentru aceasta s-au luat în considerare cîteva moduri de transport urban pentru pasageri, şi anume : transportul pe căi ferate, căile ferate din metropolă, reţeaua de trafic şi reglarea interacţiunilor. Sistemul liniilor de metrou a fost utilizat pentru optimizarea parametrilor traficului planificat.

Lucrarea 11.6/A-4 prezintă un sistem de planificare asistată de calculator pentru reţelele de transport public. Pentru început se face descrierea generală a sistemului, format dintr-un sistem de planificare a reţelei şi dintr-un sistem asistat de calculator. Pentru planificare sînt necesare date socio-economice în legătură cu zona studiată, date privitoare la reţelele de transport şi date în legătură cu cerinţele de trafic. Pe baza acestor date se vor realiza mai multe

planuri, furnizîndu-se astfel mai multe alternative pentru planificare. Un astfel de plan este definit de configurația rețelei (rute, poziționarea stațiilor de autobuz etc.), de timpul necesar pentru parcurgerea fiecărei rute, de timpul de transfer, de frecvența serviciilor și de numărul de călători. După formularea planurilor alternative, urmează predictarea fluxului de trafic pe rețeaua de transport luată în considerare. Pentru aceasta se predictează comportamentul unui călător, iar estimarea volumului de trafic pe fiecare linie a rețelei de transport se va face prin compunerea predicțiilor comportamentelor individuale. Modelul de predicție utilizat în acest studiu poate fi foarte util pentru planificarea unei rețele unimodale. În continuare se face evaluarea planurilor alternative. Dacă este necesară modificarea acestor planuri sau adăugarea unor planuri noi, se va relua procesul descris mai sus pînă cînd nu mai există posibilitatea de a găsi planuri alternative mai bune.

Sistemul asistat de calculator este dotat cu facilități de gestiune a bazei de date, facilități de gestiune grafică, facilități de gestiune a programelor de aplicație și facilități de conducere interactivă. Sistemul asistat de calculator cuprinde un sistem gazdă și un sistem satelit. Sistemul satelit este dotat cu un microcalculator cu display pe bază de tub catodic, care este conectat la calculatorul gazdă împreună cu echipamentele periferice asociate (planșetă digitală, imprimantă serială și plotter digital). Sistemul gazdă construiește baza de date, memorează programele, realizează un volum mare de calcule, iar cifrele și tabelele obținute sînt furnizate ca ieșiri.

Descrierea sistemului de planificare asistat de calculator pune în evidență faptul că, pe baza unei metode interactive om-mașină, se poate asigura gestiunea unei cantități mari de date și a mai multor tipuri de programe. Ca urmare, se poate aloca mai mult timp pentru planificare ceea ce înseamnă că este posibilă evaluarea mai multor planuri alternative, deci planificarea este îmbunătățită.

În lucrare se face prezentarea unei aplicații a sistemului într-o zonă urbană reală. S-a avut în vedere evaluarea influenței pe care a avut-o construcția metroului asupra sistemului de rețele ale transportului urban.

Probleme vizate de lucrările *11.6/A-3*, *11.6/A-5* și *11.6/A-6* se referă la aspecte care intervin în traficul urban fără a avea o generalitate atît de largă ca problemele abordate de lucrările prezentate pînă acum și care tratau aspecte legate de îmbunătățirea pe ansamblu a traficului urban.

Astfel *lucrarea 11.6/A-3* urmărește îmbunătățirea traficului pe o arteră de circulație cu sens unic prin conducerea corelată a semafoarelor care reglează circulația la fiecare joncțiune de trafic de pe artera respectivă. Pentru aceasta se propune utilizarea unui regulator fazic în logică „fuzzy" (logică infinit valentă) pentru reglarea a două joncțiuni de trafic succesive, aflate pe aceeași arteră de circulație, în condițiile în care joncțiunea de trafic este străbătută de un număr relativ mare de vehicule. Folosirea acestui regulator este contraindicată în situațiile în care numărul de vehicule este foarte mare sau foarte mic. Această problemă este soluționată prin utilizarea unui regulator în logică „fuzzy" în cooperare cu un regulator fazic în logică „fuzzy", cooperarea celor două regulatoare asigurînd o bună reglare a circulației și în cazurile în care intervin modificări în limite largi ale numărului de mașini, așa cum s-a constatat că se întîmplă înainte și imediat după orele de vîrf. Din acest motiv, reglarea semafoarelor de circulație instalate pe o arteră de circulație cu sens unic se va face prin cooperarea celor două tipuri de regulatoare, așa cum s-a arătat mai sus. Alternarea regulatoarelor se va realiza cu ajutorul unei metode de interschimbare bazată pe un algoritm de tip „fuzzy".

Pentru început, în lucrare se face prezentarea generală a unui regulator în logică „fuzzy", după care se arată modul în care acesta este utilizat în cazul unor joncțiuni succesive de trafic. Urmează prezentarea generală a regulatorului fazic în logică „fuzzy", după care se face descrierea modului în care cele două regulatoare se utilizează combinat.

În lucrare se prezintă un exemplu în care intervin doar două joncțiuni de trafic, însă metoda de interschimbare utilizată poate fi aplicată cu ușurință pentru un număr mare de joncțiuni de trafic de pe o arteră de circulație. Extinderea se face în felul următor : la prima joncțiune de trafic se utilizează un regulator în logică „fuzzy" ; fiecare dintre următoarele joncțiuni de trafic poate fi controlată printr-o cooperare dintre un regulator în logică „fuzzy" și un regulator fazic în logică „fuzzy". Algoritmul prezentat în lucrare este foarte simplu, ceea ce permite utilizarea unor microcalculatoare ieftine pentru realizarea sistemelor de reglare a semafoarelor.

Lucrarea 11.6/A prezintă un regulator ierarhizat destinat unui vehicul automatizat pentru drumuri mari. În lucrare este propusă o structură de conducere ierarhizată pe două nivele,

cu supervizor unic la nivel superior și cu 3 subordonări la nivel inferior. Configurația a fost implementată și evaluată pe baza unei simulări de laborator pentru care s-au construit în mod empiric modelele dinamicii laterale și longitudinale ale vehiculului.

O soluție eficientă pentru descongestionarea traficului rutier constă în realizarea unor rețele de circulație care să cuprindă și linii automatizate. Aceasta asigură o serie de avantaje legate de îmbunătățirea securității traficului, utilizarea mai eficientă a șoselelor, economisirea de carburanți, reducerea stressului șoferului etc. Conducerea unui astfel de sistem este realizată pe baza unei configurații ierarhizate pe 4 nivele. Un regulator centralizat asigură supervizarea tuturor operațiilor din rețea. Al doilea nivel de conducere va fi la nivel zonal. Fiecare regulator zonal va superviza activitatea într-un anumit număr de sectoare și va conduce vehiculele din sectoarele respective pe baza comenzilor furnizate către calculatoarele de sector, care formează al treilea nivel de conducere. Fiecare calculator de sector va controla vehiculele din sectorul său. Al patrulea nivel de conducere este cel corespunzător fiecărui vehicul.

Lucrarea prezintă un regulator ierarhizat utilizat pentru conducerea unui vehicul automatizat. Principalul scop urmărit a fost acela de a îmbunătăți securitatea, eficiența și fiabilitatea unui astfel de regulator. Pentru aceasta s-a utilizat o configurație multiprocesor ierarhizată pe două nivele, avîndu-se în vedere asigurarea redundanței la conversia datelor. Toate funcțiile vehiculului vor fi supervizate de calculatorul de la nivelul superior, care va transmite comenzi spre calculatoarele de sector și va utiliza la rîndul său datele prelucrate furnizate de calculatoarele de sector. Calculatoarele de sector sînt dependente de aplicație. Astfel, calculatorul 1 asigură reglarea laterală, calculatorul 2 reglează viteza, iar calculatorul 3 supraveghează defecțiunile. Dacă este necesar, la nivelul 2 se mai pot adăuga și alte calculatoare.

Funcțiile de supervizare ale calculatorului de la nivelul superior și funcțiile de conducere ale calculatoarelor de la nivelul inferior necesită numai 10 ms pentru fiecare perioadă de eșantionare de 150 ms. Astfel 92 % din timpul fiecărui calculator poate fi utilizat pentru realizarea altor funcții.

O problemă de generalitate mai largă decît cele abordate în ultimele două lucrări prezentate este tratată în *lucrarea 11.6/A-5*, care se ocupă cu problema dirijării direcției de circulație. Scopul urmărit constă în indicarea unei rute cît mai convenabile pentru un vehicul care se apropie de intersecție. Vehiculul trebuie să indice punctul de destinație, iar calculatorul local de la intersecție va răspunde prin specificarea rutei care trebuie urmărită.

Toate metodele de dirijare a direcției de circulație se bazează pe utilizarea unui tabel de dirijare a circulației. Acest tabel poate fi static (poate fi modificat numai în împrejurări critice, pe baza unei comenzi centrale), sau poate fi dinamic (modificarea tabelului se face în funcție de trafic). Dirijarea dinamică a circulației permite o rezolvare mai rapidă a situațiilor critice și o manipulare mai bună a modificărilor în circulație.

Implementarea sistemului de dirijare a circulației ridică probleme foarte mari din punct de vedere practic, problemele teoretice nefiind încă complet soluționate. Lucrarea abordează cîteva dintre problemele teoretice care nu au fost încă rezolvate. Se face o analiză a influențelor reciproce dintre rezultatele descentralizării, optimizării și prelucrărilor complexe.

Descentralizarea presupune utilizarea mai multor regulatoare în sistemul dat, fiecare regulator avînd rolul de a măsura numai anumite ieșiri, asigurînd reglarea unui număr restrîns de intrări. Această soluție este mai convenabilă deoarece, în cazul dat, centralizarea tuturor intrărilor și ieșirilor duce la scăderea fiabilității și la ridicarea costurilor.

Se utilizează un model în cadrul căruia întîrzierilor datorite congestionării circulației li se vor asocia cozi de așteptare în nodurile rețelei. Circulația revine la efectuarea unei mișcări cu viteză constantă între noduri. Timpul de trecere de la un nod la altul este dat de suma dintre întîrzierea constantă de circulație și întîrzierea datorată timpului de așteptare în coada formată la nodul de rețea. Acest model evită dinamica neliniară care ar rezulta dacă s-ar lua în considerare un profil de viteză dependent de densitatea de trafic.

Selectarea soluției optimale se face avînd în vedere comportarea pe ansamblu a sistemului. Opțiunea făcută trebuie să țină cont de cheltuielile posibile și de implicațiile unor noi sarcini de calcul care se impun în cazul în care ar fi necesară adăugarea unor noi canale de informare.

O altă problemă importantă se referă la complexitatea efortului de calcul impus de obținerea unei soluții optimale. Se prezintă o metodă de optimizare ierarhizată pentru rezolvarea problemei de reglare optimală globală, ținîndu-se seama de restricțiile legate de lungimea cozilor de așteptare în noduri și de întîrzierile din sistem. Deși metoda asigură anumite avantaje din punct de vedere numeric, este totuși prea lentă pentru lucrul în timp real.

3. SISTEME PENTRU TRANSPORTUL NAVAL

Lucrarea 11.6/B-2 abordează cîteva aspecte în legătură cu proiectarea unui pilot automat (autopilot) adaptiv pentru nave. Modelul matematic al autopilotului descris este dezvoltat pe baza unor principii de natură fizică și hidrodinamică. Autopilotul bazat pe un astfel de model prezintă o serie de avantaje față de autopilotul bazat pe modelele cutiei negre, și anume:

— permite estimarea apriorică a parametrilor pe baza unor legi fizice și hidrodinamice;

— pentru un anumit ordin al modelului, numărul parametrilor care trebuie estimați este mai mic;

— autopilotul descris în lucrare asigură minimizarea acțiunilor de cirmă nedorite, utilizînd pentru aceasta un model al mișcării de deviație induse de valuri.

Modelul matematic al autopilotului este utilizat într-o structură de conducere adaptivă. Pentru estimarea stării se va utiliza un filtru Kalman cu amplificare constantă. Matricea de amplificare a filtrului Kalman este calculată o singură dată la început, după care rămîne neschimbată. Matricea de amplificare a filtrului poate fi modificată pentru o anumită navă pe baza datelor cunoscute în legătură cu noua lungime a navei. Calculul matricii de amplificare se poate face cu o precizie suficient de mare utilizînd amplificarea filtrului corespunzătoare unei nave de lungime egală cu unitatea. Pentru implementare s-a utilizat versiunea discretizată a filtrului.

Algoritmului de estimare a parametrilor se bazează pe calculul erorii medii pătratice de predicție a filtrului Kalman. După inițializare, funcționarea algoritmului se caracterizează prin „uitarea" unei cantități de informație (referitoare la diferiți parametri) egală cu cantitatea de informație furnizată de măsurările efectuate la un moment dat. Pentru analiza acestui algoritm s-au făcut simulări pe calculator și au fost realizate experimente la bordul navei de coastă norvegiene MIDNATSOL. Testările au fost făcute în timpul operațiilor normale ale navei, iar rezultatele obținute au fost bune. Pilotul automat adaptiv va fi introdus în cel mai scurt în producția comercială.

Lucrarea 11.6/B-3 prezintă un sistem pentru conducerea încărcării unei nave cu pernă de aer. Pentru aceasta se utilizează un sistem cu ventilator activ. Principalul element al sistemului este reprezentat de fluxul axial de ridicare al ventilatorului. Caracteristicile acestuia pot fi modificate prin intermediul paletelor ale căror unghiuri de înclinare față de planul de rotație sînt modificate continuu pe durata funcționării. Mișcarea unghiulară a paletelor este obținută cu ajutorul unui element de execuție hidraulic, care la rîndul său este reglat cu ajutorul unei bucle cu reacție după poziție. Reglarea unghiurilor paletelor ventilatorului și reglarea accelerației de ridicare se realizează prin intermediul buclei interioare și respectiv prin intermediul buclei exterioare.

Funcțiile de transfer asociate dinamicii pernei de sprijin s-au obținut pe baza identificării parametrilor cu ajutorul unor modele ale planului coeficienților. Coeficienții, la rîndul lor, au fost estimați cu ajutorul unui algoritm de optimizare nelinear.

Experimentele efectuate au pus în evidență faptul că sistemul asigură un mijloc rapid și eficient de reglare a accelerației de ridicare și, în plus, este asigurată o reducere importantă a efectului de frînare în timpul traversării valurilor. Rezultatele obținute nu sînt optimale, însă se încearcă în continuare îmbunătățirea lor.

4. CONCLUZII

Lucrările prezentate în cadrul secțiunii 11.6 au tratat probleme legate de transportul terestru și naval, oferind diferite soluții pentru îmbunătățirea traficului.

Cîteva dintre lucrările prezentate au abordat probleme care privesc rezolvarea traficului urban pe ansamblu. Astfel *lucrarea 11.6/A-4* propune un sistem de planificare asistată de calculator pentru rețelele de transport public, *lucrarea 11.6/A-5* propune un sistem de dirijare a circulației, *lucrarea 11.6/A-1* propune un sistem de conducere ierarhizată descentralizată a semafoarelor de circulație, iar *lucrarea 11.6/A-2* propune un sistem de modelare pentru procesele de conducere a sistemelor de transport.

Probleme de o mai mică amploare sînt tratate în cadrul *lucrărilor 11.6/A-3* și *11.6/A-6*, care prezintă un regulator fazic în logică infinit valentă pentru conducerea sincronizată a joncțiunilor de trafic de pe arterele de circulație cu sens unic, respectiv un regulator ierarhizat pentru un vehicul automatizat.

În cadrul *secțiunii 11.6/B* au fost abordate probleme legate de transportul naval. Astfel *lucrarea 11.6/B-2* prezintă aspecte legate de proiectarea unui pilot automat adaptiv, iar *lucrarea 11.6/B-3* prezintă aspecte legate de automatizarea încărcării unei ambarcațiuni cu pernă de aer.

HIDROLOGIA, PROTECȚIA CALITĂȚII AERULUI ȘI A APEI, GOSPODĂRIREA RESURSELOR DE APĂ *

Ioan Dima, Violeta Vișan
I.C.P.G.A.
Viorel Al. Stănescu
I.M.H.

Gospodărirea apelor și protecția mediului, condiții intrinseci ale progresului economic și ale creșterii calității vieții, intervin tot mai pregnant ca o componentă majoră în activitatea cotidiană precum și în planurile sau programele de dezvoltare, ca și în studii de prospectare a viitorului pe diferite orizonturi de timp.

Prin specificul lor multi și interdisciplinar, și dată fiind complexitatea și diversitatea problemelor de gospodărire a apelor și de protecție a mediului, rezolvarea lor reclamă eforturi conjugate din partea specialiștilor din practic toate sectoarele științei și tehnologiei.

Faptul că mare parte din problemele de gospodărire a apelor și de protecție a mediului, indiferent de aria aferentă și de scara de timp, implică luarea de decizii ierarhizate, în a căror pregătire intervine prin conexiune inversă rezultatul aplicării deciziei anterioare, justificînd utilizarea conceptelor și metodelor teoriei reglării (controlului) optimale sau adaptive, își găsește confirmare în numeroasele referate prezentate pe aceste teme la congres.

În cadrul *colocviilor 11.7* și *11.8* (*secțiunile 11.8/A, 11.8/B, 11.8/C*) s-au abordat aspecte privnd hidrometeorologia, protecția calității aerului și apei și gospodărirea resurselor de apă.

Cele 18 referate prezentate se pot grupa în funcție de aspectele tratate astfel :

— aspecte de hidrometeorologie și protecția calității aerului, un număr de 3 referate [1, 2, 3] ;

— aspecte de protecție a calității apelor inclusiv epurarea apelor uzate, un număr de 7 referate [4, 5, 6, 7, 8, 9, 10];

— aspecte de gospodărire a resurselor de apă în bazine amenajate, inclusiv de alimentare cu apă, un număr de 8 referate [11, 12, 13, 14, 15, 16, 17, 18].

În cele ce urmează se prezintă, pe cele 3 subgrupe, o sinteză conținînd elementele de esență ale referatelor, din cadrul secțiunilor simpozionului interesînd domeniul apelor.

1. PERFECȚIONAREA MODELĂRII ÎN HIDROLOGIE ȘI PROTECȚIA CALITĂȚII AERULUI

Referatele din această subgrupă privesc optimizarea proiectării rețelelor de supraveghere a calității aerului [1], analiza consistenței și criterii de calibrare a parametrilor modelelor matematice în hidrologie [2]. În referatul [3] se expun rezultate din experiența specialiștilor din R.P.U. în utilizarea modelelor recursive la elaborarea prognozei operative a debitelor pe fluviul Dunărea.

1.1. Proiectarea rețelelor de supraveghere pentru controlul poluării aerului [1]

Supravegherea continuă a poluării aerului este necesară atît în scopul cunoașterii evoluției calității acestuia, cît și pentru luarea unor decizii de reglare asupra surselor de poluare. Aceasta, în condițiile în care, pe de o parte, fondurile alocate acestei activități sînt limitate, iar pe de altă parte, modelarea fizico-matematică a dispersiei poluanților în zonele urbane nu

* Ioana Dima și Violeta Vișan, I.C.P.G.A. ; Viorel Al. Stănescu, I.M.H. (coordonatori). Recenzii pe paragrafe : D. Mihu [1.1], V. Al. Stănescu [1.2] și I. P. Zlate [1.3], I.M.H. Ioan Dima [2.1, 2.2, 2.6, 2.7, 3.1, 3.4, 3.6], C. Al. Negulescu [2.3], Th. Ognean [2.4, 2.5], V. Vișan [3.2], V. Vișan și Carmen Călătan [3.3], R. Drăgășanu [3.5], V. Rojanschi [3.7, 3.8], I.C.P.G.A., Institutul de cercetări și proiectări pentru gospodărirea apelor, București. I.M.H. — Institutul de meteorologie și hidrologie, București.

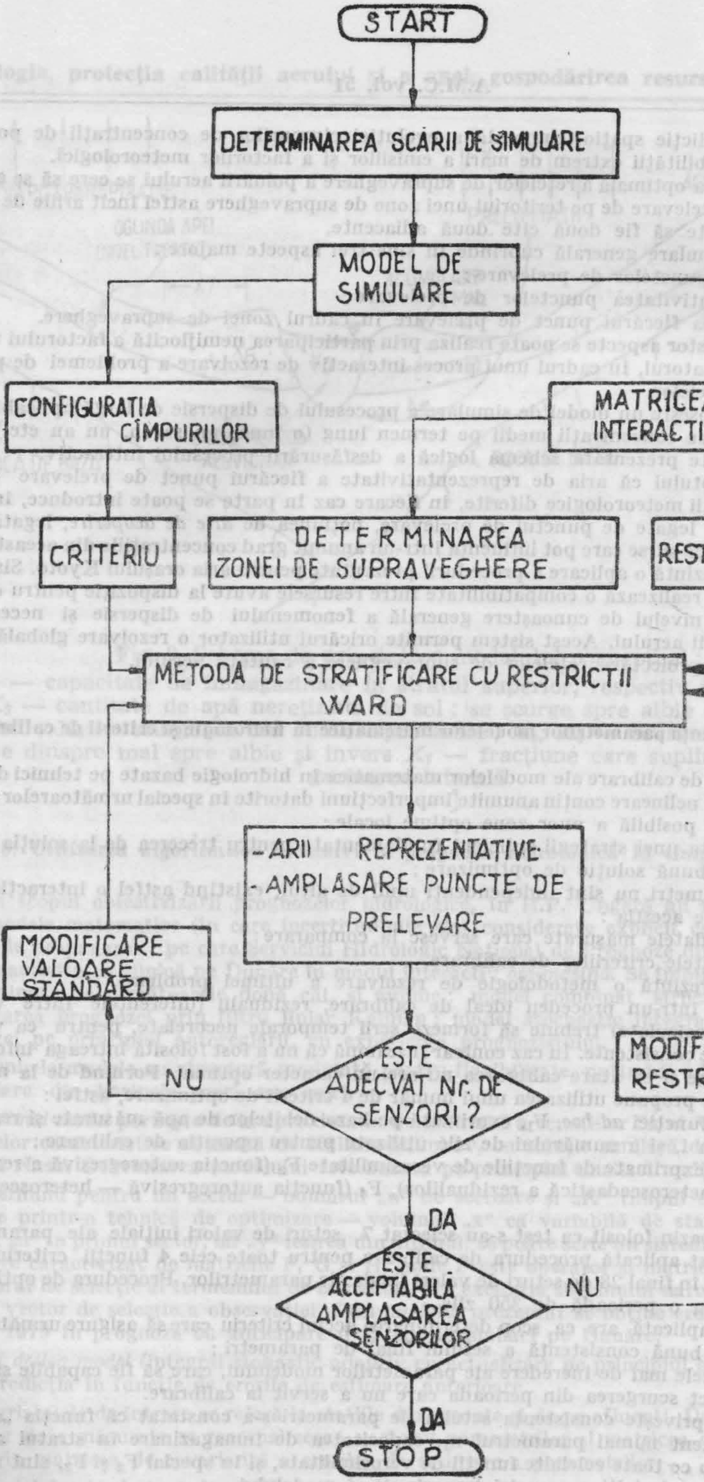


Fig. 1. Schema logică a procesului interactiv de proiectare a rețelelor de supraveghere.

poate oferi o predicţie spaţio-temporală a evoluţiei cîmpurilor de concentraţii de poluanţi ca urmare a variabilităţii extrem de mari a emisiilor şi a factorilor meteorologici.

În proiectarea optimală a reţelelor de supraveghere a poluării aerului se cere să se amplaseze punctele de prelevare de pe teritoriul unei zone de supraveghere astfel încît ariile de reprezentativitate oferite să fie două cîte două adiacente.

Această formulare generală cuprinde în sine trei aspecte majore :

— numărul punctelor de prelevare ;

— reprezentativitatea punctelor de prelevare ;

— amplasarea fiecărui punct de prelevare în cadrul zonei de supraveghere.

Tratarea acestor aspecte se poate realiza prin participarea nemijlocită a factorului uman, în dialog cu calculatorul, în cadrul unui proces interactiv de rezolvare a problemei de proiectare optimă a reţelelor.

Lucrarea foloseşte un model de simulare a procesului de dispersie de largă aplicabilitate, capabil să evalueze concentraţii medii pe termen lung (o lună, un sezon, un an etc.).

În fig. 1 este prezentată schema logică a desfăşurării procesului interactiv.

Datorită faptului că aria de reprezentativitate a fiecărui punct de prelevare devine variabilă în condiţii meteorologice diferite, în fiecare caz în parte se poate introduce, în locul reprezentativităţii legate de punctul de prelevare, noţiunea de *arie de acoperire*, legată de o sursă sau un grup de surse care pot influenţa într-un anumit grad concentraţiile din această arie.

Articolul prezintă o aplicare a procedurii prezentate pentru aria oraşului Kyoto. Sistemul interactiv propus realizează o compatibilitate între resursele avute la dispoziţie pentru o reţea de supraveghere, nivelul de cunoaştere generală a fenomenului de dispersie şi necesitatea conservării calităţii aerului. Acest sistem permite oricărui utilizator o rezolvare globală şi rapidă a problemei proiectării reţelelor de supraveghere a calităţii aerului.

1.2. Consistenţa parametrilor modelelor matematice în hidrologie şi criterii de calibrare [2]

Procedurile de calibrare ale modelelor matematice în hidrologie bazate pe tehnici de optimizare a funcţiilor nelineare conţin anumite imperfecţiuni datorite în special următoarelor cauze :

— existenţa posibilă a unor zone optime locale ;

— inexistenţa unei strategii suficient fundamentate pentru trecerea de la soluţia inferioară la cea mai bună soluţie de optimizare ;

— unii parametri nu sînt independenţi unul de altul, existînd astfel o interacţiune de natură fizică între aceştia :

— erori în datele măsurate care servesc la comparare ;

— insuficienţele criteriilor de calibrare.

Lucrarea prezintă o metodologie de rezolvare a ultimei probleme.

În general, într-un procedeu ideal de calibrare, rezidualii (diferenţele între valorile măsurate şi cele calculate) trebuie să formeze serii temporale necorelate, pentru ca valorile parametrilor să fie consistente. În caz contrar înseamnă că nu a fost folosită întreaga informaţie de care se dispune şi ca atare calibrarea nu are un caracter optimal. Pornind de la modelul descris în fig. 2 se propune utilizarea unui număr de 4 criterii de optimizare, astfel :

— criteriul funcţiei *ad-hoc*, F_1, exprimată pe baza debitelor de apă măsurate şi respectiv calculate din ziua i, şi a numărului de zile utilizate pentru operaţia de calibrare ;

— criteriile exprimate de funcţiile de verosimilitate F_2 (funcţia autoregresivă a rezidualilor), F_3 (funcţia heteroscedastică a rezidualilor), F_4 (funcţia autoregresivă — heteroscedastică a rezidualilor).

Pentru un bazin folosit ca test s-au selectat 7 seturi de valori iniţiale ale parametrilor modelului şi a fost aplicată procedura de calibrare pentru toate cele 4 funcţii criteriu menţionate, rezultînd în final 28 de seturi de valori finale ale parametrilor. Procedura de optimizare s-a aplicat pentru o perioadă de 350 zile.

Procedura aplicată are ca scop determinarea acelui criteriu care să asigure următoarele :

— cea mai bună consistenţă a setului final de parametri ;

— valorile cele mai de încredere ale parametrilor modelului, care să fie capabile să simuleze cît mai corect scurgerea din perioada care nu a servit la calibrare.

În ceea ce priveşte consistenţa setului de parametri s-a constatat că funcţia „ad-hoc" estimează consistent numai parametrul x_1, capacitatea de înmagazinare în stratul superior al solului, în timp ce toate celelalte funcţii de verosimilitate, şi în special F_2 şi F_3, sînt mai eficace în estimarea celorlalţi parametri importanţi ai modelului.

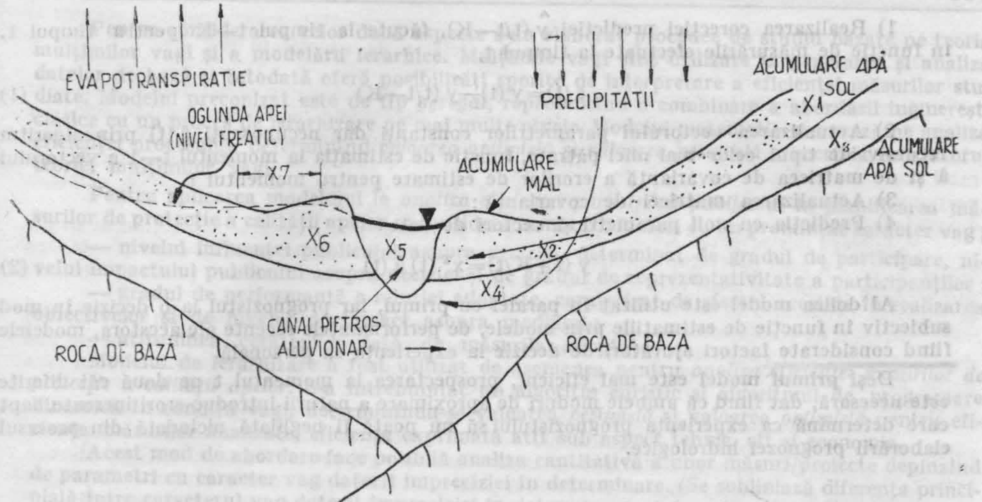


Fig. 2. Schema de principiu a modelului conceptual :

X_1, X_3 — capacitate de înmagazinare în stratul superior, respectiv inferior al solului ; X_2 — cantitate de apă nereţinută de sol ; se scurge spre albie şi se infiltrează în subsol ; X_4 — scurgerea din subsol spre albie ; X_5, X_6 — scurgere, respectiv infiltraţie dinspre mal spre albie şi invers X_7 — fracţiune care suplimentează înmagazinarea laterală

1.3. Utilizarea algoritmilor recursivi la prognoza hidrologică în timp real [3]

În scopul obiectivizării prognozelor hidrologice, în R.P. Ungară au fost elaborate [3] două modele matematice (în care incertitudinile sînt considerate explicit ca tip, mărime şi şi mod de propagare), pe care Serviciul Hidrologic Naţional de Prognoze din Ungaria le foloseşte în activitatea zilnică pe Dunăre in modul interactiv *om-maşină*. Se îmbină astfel informaţiile obţinute pe următoarele două căi, şi anume : model combinat structural-stohastic cu actualizarea prognozei prin filtru liniar Kalman ; model stochastic integral cu actualizarea prognozei pe principiul autoreglării, cu experienţa prognozistului.

Ambele modele au structură modulată pentru flexibilitate, putînd fi adaptate condiţiilor particulare din bazinele mari sau mici.

Primul model porneşte de la aproximaţia de ordinul I a ecuaţiilor Saint-Venant, sub forma sectoarelor caracteristice obţinută de Kalinin-Miliukov, cu soluţie similară de altfel modelului cascadei Nash. Dezvoltarea modelului este bazată pe principiul analizei mulţimii stărilor.

Definind pentru un sector — numărul „n" de sectoare şi „K" timpul de propagare sînt obţinute printr-o tehnică de optimizare — volumul „x" ca variabilă de stare şi considerînd intrarea „u" în primul sector, iar „y" ieşirea din ultimul, se poate scrie un sistem dinamic de ecuaţii lineare caracterizat de matricile F, G şi H, unde F este operator şi matrice de selectare, G — vectorul de selecţie al termenului de comandă, sau excitaţia sistemului natural, iar H — operator şi vector de selecţie a observaţiei. Prin modelul prezentat se obţine eroarea de 13 cm pe 10 mai 1979 în prognoza cu anticipare de 5 zile la Mohaci pe Dunăre.

Al doilea model (integral stohastic adaptiv cu actualizare pe principiul autoreglării) realizează predicţia în funcţie de erorile de estimare anterioare.

Variabilele de intrare se referă la staţiile din amonte şi de pe afluenţi. După recepţionarea ultimelor date măsurate se reactualizează vectorul parametrilor şi matricea de pondere exprimată prin matricea de covarianţă a erorilor de estimare a parametrilor.

Modelul este derivat din ARMA şi realizează prognoza în 4 paşi :

1) Realizarea corecţiei predicţiei $\hat{y}$ (t/t—K), făcute la timpul t—K pentru timpul t, în funcţie de măsurările efectuate la timpul t,

$$\varepsilon(t)=y(t)-\hat{y}(t/t-K) \quad (1)$$

2) Actualizarea vectorului parametrilor constanţi dar necunoscuţi $\hat{a}$ (t) prin algoritm recursiv de tipul celor mai mici pătrate, funcţie de estimaţia la momentul t—1 a vectorului $\hat{a}$ şi de matricea de covarianţă a erorilor de estimare pentru momentul t.

3) Actualizarea matricei de covarianţă;

4) Predicţia cu noii parametri şi vectori de date:

$$\hat{y}(t+K/t)=x^T(t)\,\hat{a}(t) \quad (2)$$

Al doilea model este utilizat în paralel cu primul, iar prognozistul ia o decizie în mod subiectiv în funcţie de estimaţiile prin modele, de performanţele recente ale acestora, modelele fiind considerate factori ajutători de decizie la experienţa sa personală.

Deşi primul model este mai eficient, prospectarea la momentul t pe două căi diferite este necesară, dat fiind că ambele moduri de aproximare a naturii introduc erori inerente, fapt care determină ca experienţa prognozistului să nu poată fi neglijată niciodată din procesul elaborării prognozei hidrologice.

2. ABORDĂRI MAI EFICIENTE ÎN CONCEPEREA ŞI APLICAREA MĂSURILOR DE PROTECŢIE A CALITĂŢII APELOR

În cadrul acestei grupe de referate sînt expuse rezultate şi abordări în curs pentru îmbunătăţirea şi extinderea aplicării conceptelor, metodelor şi procedeelor bazate pe teoria mulţimilor vagi şi pe teoria reglării optimale în activitatea practică de supraveghere şi protecţie a calităţii apelor, precum şi în exploatarea curentă a staţiilor de epurare.

Rezultatele şi preocupările legate de utilizarea teoriei mulţimilor vagi şi a modelelor ierarhizate în analiza şi luarea deciziilor de protecţie a calităţii apelor fac obiectul referatului [4].

Aspecte legate de folosirea unui *estimator de stare* pentru supravegherea poluării apelor unui rîu avînd regim de curgere nestaţionar sînt redate în referatul [5].

Referatul [6] tratează optimizarea reţelelor de canalizare prin controlul funcţionării pe mai multe nivele.

În referatele [7, 8] se expun aspecte legate de aplicarea modelării matematice la procesele de epurare a apelor prin procedeul cu nămol activat, respectiv în cazul procesului de neutralizare a apelor uzate.

Prezentarea unui sistem de dirijare a funcţionării unei staţii de epurare prin utilizarea unor scenarii de decizii face obiectul referatului [9].

Referatul [10] analizează critic stadiul aplicării teoriei reglării optimale la problemele de protecţie a calităţii apelor. Se evidenţiază posibilităţile de extindere a utilizării conceptelor acestei teorii în condiţiile favorabile ale dezvoltării tehnicii şi de accentuare a cerinţelor privind protecţia calităţii apelor, subliniindu-se totodată necesitatea unor reorientări asupra rolului exploatării staţiilor de epurare.

2.1. Utilizarea teoriei mulţimilor vagi şi a modelelor ierarhizate la protecţia calităţii apelor în cazul unor impurificări neconcentrate [4]

În studiul măsurilor privind majoritatea sistemelor tehnice este posibilă optimizarea unor parametri de performanţă, definiţi şi măsurabili, în condiţii impuse de restricţii definibile şi măsurabile.

În cazul sistemelor de gospodărire a apelor şi, în particular, al celor pentru protecţia calităţii apelor, definirea funcţiei obiectiv şi a restricţiilor este o problemă deosebit de complexă.

Astfel, analiza măsurilor de protecţie a calităţii apelor, legat de sursele de impurificare neconcentrate din centrele populate, implică aspecte care nu pot fi cuantificate sau măsurate cu precizie. Două asemenea aspecte privesc participarea populaţiei la elaborarea măsurilor şi la supravegherea calităţii apei, precum şi estimarea eficienţei tehnice şi economice a variantelor de soluţionare.

Pentru considerarea celor două aspecte s-au elaborat procedee de studiu bazate pe teoria mulţimilor vagi şi a modelării ierarhice. Mulţimile vagi sînt utilizate la generarea şi analiza datelor de bază şi totodată oferă posibilităţi sporite de interpretare a eficienţei măsurilor studiate. Modelul preconizat este de tip *agregat*, reprezentînd o combinare a abordării inginereşti clasice cu un proces de ierarhizare pe mai multe nivele. Modelul matematic se axează pe analiza eficienţei procesului, determinînd *valoarea optimistă* şi *valoarea pesimistă* a acesteia cu ajutorul teoriei mulţimilor vagi.

Pentru aplicarea modelului la *analiza eficienţei participării publicului* în planificarea măsurilor de protecţie a calităţii apelor se consideră următorii factori cu un pronunţat caracter vag ;

— nivelul influenţei publicului asupra deciziei, determinat de gradul de participare, nivelul impactului publicului asupra deciziei şi de gradul de reprezentativitate a participanţilor ;

— gradul de performanţă a soluţiei adoptate, caraterizat de efectul acesteia în realizarea obiectivelor şi de nivelul de impact asupra mediului ;

— atitudinea cetăţenilor faţă de măsurile adoptate.

Modelul de ierarhizare a fost utilizat de asemenea pentru *analiza eficienţei măsurilor de protecţie a calităţii apelor*. S-a întrebuinţat un algoritm euristic şi algoritmul de programare dinamică în condiţii vagi, determinîndu-se valoarea *pesimistă* şi valoarea *optimistă* pentru eficienţa măsurilor analizate, eficienţă exprimată atît sub aspect tehnic, cît şi economic.

Acest mod de abordare face posibilă analiza cantitativă a unor măsuri/proiecte depinzînd de parametri cu caracter vag datorit impreciziei în determinare. (Se subliniază diferenţa principială între caracterul vag datorit impreciziei în determinare şi incertitudinea datorită în principal naturii aleatoare a elementelor analizate). Valoarea *pesimistă* şi cea *optimistă* a funcţiei de eficienţă oferă planificatorului o imagine mai reală asupra limitelor cîmpului determinat de diferite atitudini privind diferiţii factori sau parametri, decît în cazul calculării unei singure valori. Se consideră de asemenea utilă determinarea unei valori intermediare, ceea ce ar ajuta la alegerea soluţiei făcînd uz şi de procedeul Delphi.

2.2. Supravegherea poluării apelor într-un rîu influenţat de fenomenul de maree, folosind un estimator de stare cu structură comutabilă [5]

Înrăutăţirea calităţii apei rîurilor datorită impurificării de natură organică este o problemă actuală şi urgentă în multe ţări industrializate. Această situaţie accentuează necesitatea realizării de sisteme de supraveghere, în scopul obţinerii de informaţii pentru decizii de intervenţie care să prevină depăşirea concentraţiei admisibile.

Preocupările de pînă acum ale specialiştilor atît privind modele de tip cu debit constant, cît şi modele de tip dinamic nu se referă la situaţii de nestaţionaritate hidromecanică.

Referatul prezintă un procedeu de determinare denumit *estimator de stare* pentru supravegherea variaţiei calităţii apelor într-un rîu cu regim de nivel variabil (sub influenţa fluxului şi refluxului mării în care deversează). Procedeul permite luarea în considerare a condiţiilor de nestaţionaritate şi de fluctuaţie a regimului curgerii apei şi este utilizabil la supravegherea impurificării organice în rîurile cu nivelul apei influenţat de fenomenul de maree, atunci cînd se dispune numai de măsurări de la senzori situaţi la mare distanţă.

Estimatorul constă din două componente de bază : *modelul procesului analizat* pentru generarea de estimări ale variabilelor de stare şi o *buclă de corecţie* prin care se semnalează abaterile dintre valorile estimate de model şi valorile determinate efectiv, reprezentînd conexiunea inversă pentru îmbunătăţirea estimărilor viitoare.

Modelul de rîu este de tip unidimensional, în care timpul este considerat ca o variabilă continuă, iar spaţiul ca variabilă cu valori discrete. În model intervin pentru un sector de rîu următoarele variabile :

— valori medii pe sector pentru secţiunea de curgere (secţiunea vie), concentraţia oxigenului dizolvat şi consumul biochimic de oxigen ;

— debitul de apă şi de impurificări efluent din sectorul considerat, în sectorul din aval.

După încercări, în care s-a utilizat *filtrul Kalman extins*, acesta a fost înlocuit cu un filtru mai simplu de tip *matrice de amplificare* folosind un microcalculator la locul de observare, în teren. Pentru a lua în considerare inversarea direcţiei de curgere a apei, matricea de amplificare din alcătuirea estimatorului are structură comutabilă.

Rezultatele simulărilor efectuate arată că acest procedeu permite determinări cu suficientă precizie în condiţii de instaţionaritate a regimului apei în rîuri.

2.3. Conducerea multinivel a exploatării rețelelor de canalizare [6]

Conducerea multinivel se referă la sistemele unitare de canalizare alcătuite din tronsoane de conductă, bazine tampon și elemente de deversare. În absența ploilor apele uzate menajere sînt epurate și apoi se descarcă în receptor. În caz de ploaie debitele din rețea depășesc considerabil capacitatea stației de epurare, rezultînd descărcarea în emisar a unei cantități de ape neepurate. În ultima perioadă s-a manifestat un interes crescînd pentru reducerea efectelor deversărilor de ape de ploaie neepurate din sistemele unitare de canalizare.

Reducerea descărcărilor de apă neepurată în emisar se poate realiza prin reținerea în bazinele tampon a unei părți din volumul de apă scurs în timpul ploii. Imediat după ploaie, apa din bazinele tampon se trece prin stația de epurare pentru ca aceste bazine să poată fi utilizate la întreaga capacitate în cazul unei alte ploi.

Sistemul de conducere multinivel combină evacuarea debitului optim pentru minimizarea deversărilor de apă neepurată în emisar cu costuri moderate de realizare. Structura pe mai multe nivele este alcătuită dintr-un *nivel de optimizare*, un *nivel de reglare directă* și un *nivel de adaptare.*

Într-o situație dată de debite din precipitații reglarea optimă a scurgerii necesită considerarea sistemului de canalizare în ansamblu. Procedura de optimizare elaborată în acest scop consideră sistemul de canalizare format din subrețele cu și, respectiv, fără capacități de stocare. *Problema de optimizare* poate fi rezolvată principial prin aplicarea algoritmului de programare dinamică, dar memoria necesară devine foarte mare dacă problema include mai mult de 3—4 variabile de stare și peste 10—15 intervale de timp. De aceea, a fost aplicat *principiul maximului discret*, care permite rezolvarea unei probleme de optimizare cu mai mult de 8 variabile de stare și peste 60 intervale de timp de calcul.

Pentru aplicarea principiului *maximului discret* a fost necesară înlăturarea unor inconveniente specifice, astfel :

— unele variabile de stare exprimate ca restricții duc la dificultăți deosebite. Pentru depășirea acestei dificultăți s-a adoptat metoda *funcțiilor de penalitate* ;

— problema de optimizare este rezolvată prin exprimarea condițiilor necesare de optimalitate și formularea acesteia ca o problemă de valori limită în 2 puncte ; soluționarea problemei astfel formulate poate conduce la un minim local în locul celui global. Dat fiind însă că *funcția obiectiv* este convexă și restricțiile sînt liniare, soluția problemei de optimizare reprezintă minimul global ;

— problema se rezolvă folosind un algoritm numeric a cărui convergență trebuie examinată. Aplicînd tehnica gradienților conjugați se garantează o convergență rapidă chiar și pentru matrici mari de penalitate.

Soluționarea problemei de optimizare necesită cunoașterea următoarelor elemente :

— debite afluente ;

— decalajul de timp determinat de tronsoanele de canalizare ;

— condiții inițiale ;

— matrici de ponderare.

Nivelul de reglare directă are ca scop modificarea comportării dinamice a procesului astfel încît să se apropie cît mai mult de modelul simplificat utilizat în nivelul de optimizare. Aceasta se realizează prin menținerea variabilelor de stare în vecinătatea traiectoriilor de referință optime determinate în cadrul nivelului de optimizare.

Nivelul de adaptare are ca sarcină :

— să asigure date bazate pe măsurări efective ;

— să inițieze o nouă analiză de optimizare ori de cîte ori datele respective au devenit necorespunzătoare.

2.4. Modelarea și reglarea procesului de epurare cu nămol activ utilizînd un model autoregresiv [7]

Pentru modelarea procesului de epurare în condiții dinamice se face în general apel la modelele matematice mecaniciste bazate pe ecuații diferențiale care exprimă bilanțurile de materiale din cadrul procesului. Asemenea modele au dezavantajul că includ o serie de coeficienți cinetici, a căror valoare se modifică o dată cu schimbarea parametrilor de funcționare ai stației

de epurare — funcţionarea staţiei în regim *dinamic.* Modelul autoregresiv elaborat îşi propune să înlăture acest dezavantaj.

Modelul autoregresiv este un model statistic din categoria modelelor „cutie neagră". Un asemenea model se defineşte în felul următor : starea prezentă a unui sistem este o sumă a combinaţiilor lineare a variabilelor sistemului din trecut şi a zgomotului alb. Aceasta se exprimă cu ajutorul ecuaţiei :

$$X(n) = \sum_{m=1}^{M} A(m)\,X(n-m) + U(n)$$

în care : A (m) este matricea coeficienţilor ;
U (n) — zgomotul alb ;
M — ordinul modelului.

Schema generală după care se aplică modelul autoregresiv cuprinde următorii paşi :
— culegerea de date dintr-o staţie de epurare reală ;
— tararea/calibrarea modelului definit prin relaţia de mai sus ;
— evaluarea (calculul contribuţiei zgomotului şi predicţia) ;
— obţinerea de date din instalaţia pilot ;
— calibrarea modelului pentru analize de sistem şi proiectarea sistemului de reglare.

Modelul autoregresiv este capabil să reproducă datele obţinute în determinările experimentale din staţia de epurare. Prin utilizarea lui în cadrul unei staţii pilot s-a putut prevedea modul de schimbare a variabilelor de operare a procesului. Modelul se poate implementa într-o staţie de epurare propriu-zisă folosindu-se metoda reglării linear pătratice.

2.5. Reglarea adaptivă a unui proces de neutralizare a apelor uzate [8]

Se prezintă un procedeu nou pentru reglarea adaptivă a pH-ului în procesul continuu de neutralizare dintr-o instalaţie de epurare a apelor.

Ca urmare a limitărilor de spaţiu şi a altor considerente impuse de procesul tehnologic, neutralizarea are loc într-un mic reactor cu timp de staţionare mic, de ordinul a 30 s.

Comportarea statică a procesului de neutralizare este determinată prin curbele de titrare a apei uzate afluente. Compoziţia şi concentraţia apelor uzate şi deci şi curbele lor de titrare variază foarte repede. Variaţia rapidă a parametrilor de proces şi marile perturbări ale procesului fac ca mijloacele de reglare convenţionale să nu mai fie satisfăcătoare. Pentru a depăşi aceste dificultăţi s-a apelat la conceptul de *reglare adaptivă*, în care bucla (ciclul) principală de reglare este condusă de o buclă suplimentară de reglare adaptivă.

Bucla de reglare adaptivă manevrează rezultatul/avansul algoritmului de reglare principal astfel încît în cazul unor abateri, pH-ul efluentului să se apropie de punctul de neutralizare prin urmărirea **traiectoriei de referinţă** prescrise, care constituie criteriul de adaptare. Prin acest mod de adaptare, orice gen de abatere poate fi controlată/corectată cu suficientă rapiditate.

Sistemul de reglare este constituit din următoarele blocuri :
— algoritmul de reglare ;
— viteza efectivă a abaterilor de reglare ;
— calculul vitezei de referinţă ;
— corecţia globală.

Sistemul de reglare adaptivă a fost testat într-o staţie pilot. În acest scop a fost realizată o staţie pilot de neutralizare cu un mic nou reactor cu o capacitate de 1,5 m^3/h şi timp mediu de staţionare de 25 s.

Sistemul de reglare adaptivă este implementat pe un minicalculator şi numeroase experienţe au arătat utilitatea sa pentru o largă gamă de abateri.

În staţia centrală de epurare sistemul se va implementa pe un microprocesor de reglare. Se subliniază pe cazul prezentat că aplicarea metodelor moderne de reglare în procesele inginereşti poate conduce la importante beneficii.

2.6. Sistem bazat pe cunoaştere pentru conducerea procesului de epurare a apelor uzate [9]

Procesul de epurare a apelor uzate la o staţie de epurare existentă poate fi caracterizat prin reglarea concentraţiei oxigenului dizolvat (OD), a concentraţiei materiilor în suspensie (MTS) şi a cantităţii de nămol cu un calculator numeric.

Sistemul bazat pe cunoaştere propus cuprinde trei părţi:
— *baza de date* ;
— *interpretor* ;
— *memorie*.

Setul regulilor de operare se referă la :
1) Identificarea situaţiei ;
2) Planificare şi control ;
3) Diagnoză şi procedee de tratare ;
4) Instrucţiuni de exploatare.

Spre exemplificare, pe baza experienţei în exploatarea bazinelor cu nămol activ la scară naturală pe mai mulţi ani, se definesc şase situaţii tipice clasificate ca :
— situaţia optimă ;
— 4 situaţii suboptime ;
— situaţia anormală.

Aceste situaţii sînt evaluate pe baza timpului de retenţie al solidelor (TRS), consumului biochimic de oxigen (CBO) şi temperaturii, răspunsurile fiind date de operator. Cel care evaluează situaţia (optimă, suboptimă, anormală) decide asupra operaţiilor necesare ca urmare a perturbărilor datorate încărcării organice.

În mod asemănător poate fi evaluată situaţia funcţionării bazinelor de fermentare a nămolului, a platformelor de deshidratare, a staţiilor de pompare etc. Din inventarul datelor de bază furnizate de operator se deduc modificările care trebuie făcute în funcţionarea instalaţiei pe baza răspunsurilor furnizate automat de calculator, care are în memorie un număr mare de instrucţiuni.

Sistemul expert propus poate ajuta la rezolvarea problemelor ivite în exploatarea staţiilor de epurare, în luarea deciziilor pentru îmbunătăţirea funcţionării. Avantajul sistemului derivă din faptul că datele de bază sînt furnizate pe baza experienţei operatorului, deciziile fiind luate de calculator.

Preocupările în continuare au în vedere îmbunătăţirea performanţelor şi sporirea încrederii în acest sistem, de exemplu prin determinarea automată a regulilor de exploatare.

2.7. Reglare şi evaluare operativă în protecţia calităţii apelor [10]

În referat se analizează mai întîi rezultatele obţinute în aplicarea teoriei reglării la două probleme clasice de protecţie a calităţii apelor : reglarea oxigenului dizolvat şi reglarea procesului cu nămol activat în instalaţiile de epurare a apelor. Legat de aceasta se menţionează că aplicarea reglării automate nu a înregistrat succese deosebite şi s-a axat mai ales pe problema reglării valorii de referinţă în condiţiile perturbaţiilor frecvente.

În fig. 3 este redată schema conceptuală a unui sistem de reglare pentru mai mulţi indicatori. Se menţionează că simplitatea inevitabilă a figurii 3 maschează caracterul pregnant ierarhic al gospodăririi integrate a resurselor de apă, pe ansamblul bazinelor ca şi localizat, de exemplu, în cazul conducerii funcţionării reţelelor de canalizare.

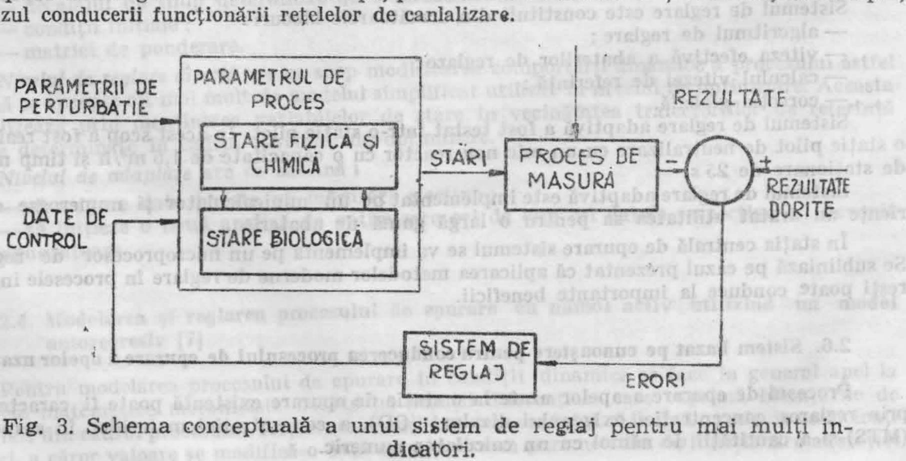


Fig. 3. Schema conceptuală a unui sistem de reglaj pentru mai mulţi indicatori.

Se consideră în general că reglarea în probleme de poluare a apelor priveşte, sau se referă la, exploatarea staţiilor de epurare şi a staţiilor de tratare a apelor. Deşi acesta este cazul general, contrubuţii importante la gospodărirea calităţii apei pot fi aduse prin utilizarea lacurilor de acumulare, a schemelor cu pompare din apa subterană ca şi a altor sisteme de exemplu centuri (perdele) din stuf.

Problema reglării oxigenului dizolvat a cunsocut progrese, inclusiv sub forma reglării dinamice a procesului. Dintre dificultăţile întîmpinate se menţionează imposibilitatea determinării *on-line* a consumului biochimic de oxigen (CBO), iar condiţiile tehnologice actuale nu permit reglarea valorii concentraţiei CBO a apelor evacuate din staţiile de epurare.

Totodată, recent a fost remarcat faptul că preocupările anterioare nu au luat în considerare gradul foarte înalt de incertitudine în comportarea sistemelor de resurse de apă.

În ce priveşte reglarea procesului de epurare cu nămol activat, staţiile de epurare oferă o flexibilitate în funcţionare satisfăcătoare, urmărindu-se în special reducerea concentraţiei de substanţe solubile, a CBO şi a substanţelor în suspensie din apele uzate. A devenit larg utilizabilă în practică reglarea funcţionării aeratorului în legătură cu concentraţia oxigenului dizolvat, printr-o schemă de tip cu o singură buclă închisă, folosind un regulator de tip cu *acţiune proporţională şi integrală*. De asemenea sînt preocupări în aplicarea reglării între conţinutul apei reziduale neepurate şi viteza de recirculare a nămolului.

Se evidenţiază însă posibilitatea unor rezultate necorespunzătoare în reglarea procesului cu nămol activat, ca de exemplu în cazul depunerii nămolului şi al distrugerii populaţiei de microorganisme în prezenţa unor concentraţii deosebit de mari de substanţe toxice.

Condiţiile favorabile oferite de dezvoltarea tehnicii şi de accentuarea cerinţelor privind protecţia calităţii apelor oferă posibilităţi de extindere a utilizării conceptelor teoriei reglării optimale.

Recunoaşterea variabilităţii performanţelor este fără îndoială o schimbare pozitivă în favoarea aplicării conceptelor *reglării*. Caracterizarea performanţei ca „satisfăcătoare" în sens *vag* este bine venită, complementar celei asociate utilizării cuantilei de 95%.

Dat fiind caracterul problemelor de protecţia calităţii apelor se consideră de asemenea posibilă obţinerea unor bune rezultate prin aplicarea atentă a unor metode din teoria sistemelor ierarhice. Unele din aspectele menţionate vor implica însă reorientări în considerarea rolului exploatării staţiilor de epurare.

3. PROGRESE PRIVIND APLICAREA TEHNICILOR DE ANALIZĂ A SISTEMELOR LA FUNDAMENTAREA DECIZIILOR ÎN GOSPODĂRIREA RESURSELOR DE APĂ

În cadrul acestei categorii de probleme sînt de scos în evidenţă preocupările legate de extinderea utilizării tehnicilor de analiză a sistemelor şi de perfecţionare a utilizării posibilităţilor oferite de mijloacele moderne de calcul.

Referatul [11] analizează critic unele neajunsuri privind stadiul de utilizare a tehnicilor de modelare în studiul problemelor de gospodărire a apelor şi punctează unele modalităţi de obţinere a unor modele mai robuste pentru reprezentarea fenomenului hidrologic într-un bazin; se subliniază necesitatea utilizării conceptului de *optim global* bazat pe determinarea *frontierei de negociere* în pregătirea deciziilor privind planificarea şi exploatarea amenajărilor de gospodărire a apelor.

O contribuţie vizînd extinderea posibilităţilor de interpretare fizică a *regulilor de decizie de tip linear* în exploatarea unui lac de acumulare este expusă în referatul [12].

Prezentarea unui concept de abordare de tip *anti-risc* în stabilirea regulilor de exploatare a unui lac de acumulare cu folosinţe multiple face obiectul referatului [13].

Referatul [14] scoate în evidenţă, pe un exemplu concret, posibilităţile oferite de asigurarea unor informaţii hidrologice *on-line* pentru creşterea eficienţei în exploatare a unui lac de acumulare avînd funcţiune dublă: combaterea inundaţiilor şi regularizarea debitelor în scopul producerii de energie hidroelectrică.

Aspectele legate de utilizarea *teoriei mulţimilor vagi* în probleme de alocare a resurselor fac obiectul referatului [15].

Referatul [16] prezintă un sistem de analiză bazat pe modelarea matematică şi pe folosirea unui procedeu *interactiv* (conversaţional) om-calculator, în planificarea proiectelor de amenajare în domeniul apelor.

Utilizarea modelării matematice şi a lucrului conversaţional om-calculator legate de extinderea reţelelor de alimentare cu apă şi respectiv, în exploatarea staţiilor de pompare, fac obiectul referatelor [17], respectiv [18].

3.1. Cvasi-optimalitate în modelele de gospodărire a apelor şi în cele fizice [11]

Se analizează aspectele legate de utilizarea conceptului de *cvasi-optimalitate* şi de *optim global* în domeniul gospodăririi apelor.

Sînt menţionate trei cauze principale care împiedică utilizarea pe scară largă a tehnicilor de *analiză a sistemelor* în modelele privind sistemele de gospodărire a apelor : (1) existenţa unor obiective (cerinţe) contradictorii, (2) existenţa unei *suprafeţe de răspuns* relativ plate în vecinătatea optimului global şi în consecinţă posibilitatea acceptării unei game de variante de acţiune, şi (3) caracterul imprecis al datelor şi nesiguranţa rezultatelor. Toate acestea conduc la rezultate *negociabile* şi nu la prescripţii lipsite de ambiguitate. Efectuarea unor asemenea negocieri în sectorul public nu se desfăşoară în condiţii foarte bune, astfel că uneori este mai simplu şi mai comod să se adopte procedeul „soluţiei prestabilite".

Hidrologia s-a dezvoltat ca o ramură a ingineriei, deoarece inginerii au avut dintotdeauna răspunderea pentru prevenirea efectelor inundaţiilor şi secetelor. Aceştia completau golurile în datele hidrologice, în general pe baze empirice.

Modelele convenţionale *precipitaţii-curgere* multiparametrice, elaborate în ultimii 10—15 ani, nu se dovedesc utilizabile în practică deoarece mici variaţii ale numeroaselor elemente componente conduc la modificări pronunţate ale elementului final, curgerea rezultată ; aceste modele sînt greu de calibrat şi aplicat în calculele curente de proiectare.

Astfel, dată fiind varietatea fenomenelor şi a ariilor lor se preconizează dezvoltarea de modele la scară medie care să asigure o reflectare fizică cît mai plauzibilă şi care să păstreze ceva din natura *stohastică* şi din caracterul *vag* al proceselor implicate. Cu alte cuvinte, este mai bine să lucrăm cu modele cu cîţiva parametri cît mai robuşti (care nu sînt supuşi unor modificări prea mari), decît cu modele foarte sensibile la perturbări şi la erori de calibrare.

De aceea, pentru modelarea fenomenelor într-un bazin hidrografic se preconizează utilizarea a 4 variabile de bază : precipitaţiile, înmagazinarea (în sol), evapotranspiraţia şi scurgerea.

În aplicarea tehnicilor de analiză a sistemelor la proiectarea şi exploatarea unor sisteme mari şi complexe, problema poate prezenta un mare număr de soluţii relativ apropiate din punct de vedere tehnic. Uneori, soluţii diferite de soluţia optimă sub aspect tehnic pot fi mai acceptabile din punct de vedere instituţional şi politic. Astfel barajul Hoover din S.U.A. este un triumf al tehnicii construcţiilor, dar constituie o eroare din punct de vedere al gospodăririi apelor ; trebuie însă amintit că, dată fiind situaţia de depresiune economică din acea perioadă, alegerea soluţiei s-a bazat pe criteriul bunăstării economice, respectiv pentru ocuparea forţei de muncă.

În alegerea soluţiilor, în cazul amenajărilor cu cerinţe competitive, este nevoie să se negocieze asupra obiectivelor şi a restricţiilor. Conceptul de *optim global* devine necesar deoarece decizia finală implică afectarea parţială sau totală a intereselor uneia sau alteia dintre cerinţele competitive.

Pareto-optimalitatea, ca mod de rezolvare a unor asemenea probleme, caută să identifice soluţii „de compromis", de etapă, determinînd *frontiera de negociere*, cu cîteva soluţii posibile, alegerea finală a soluţiei urmînd a se face în cadrul procesului instituţional-politic.

3.2. Analiza şi interpretarea fizică a regulilor de decizie privind regimul de exploatare a unui sistem de gospodărire a apelor [12]

În legătură cu proiectarea şi exploatarea sistemelor de gospodărire a apelor se cunosc numeroase încercări bazate pe utilizarea *regulilor de decizie lineare*.

Procedeele elaborate pînă în prezent se limitează totuşi la formularea problemei capacităţii minime a acumulării. Problema regimurilor de exploatare a unei acumulări existente pe baza unor asemenea reguli nu şi-a găsit încă aplicare.

Referatul prezintă un mod nou de abordare care permite utilizarea regulilor de decizie lineare atît la proiectare, pentru determinarea capacităţii acumulării, cît şi pentru studiul exploatării unei acumulări existente.

Pentru aceasta se formulează aşa-numita *problemă operativă*, care se rezolvă prin repetiţie pe baza informaţiilor curente şi a alegerii adecvate a formei *indicelui de performanţă*.

Ca indice de performanţă pentru un sistem din două lacuri s-a utilizat printre expresia:

$$\max_{S^*} \left(\sum S_i^{*1} + 2\sum S_i^{*2}\right)$$

reflectînd condiţia de a stoca în lac o cantitate cît mai mare de apă. În această expresie S_i^* reprezintă valoarea probabilă (valoarea medie aşteptată) a volumului de apă acumulat în lac, S_i.

Analizele efectuate pe exemple numerice au condus la următoarele concluzii:

— regulile de decizie lineară au o interpretare fizică şi permit formularea problemei de proiectare şi de exploatare ca probleme de programare lineară;

— aceste reguli sînt deosebit de utile în cazul sistemelor complexe, în care alte metode devin inaplicabile ca urmare a dimensiunilor excesiv de mari ale problemei;

— inconvenientul rezultînd din ipoteza că regulile sînt lineare poate fi neglijat, deoarece în fiecare sezon, sau interval de calcul, problema se poate rezolva repetitiv luînd în considerare condiţiile curente.

3.3. Abordare de tip anti-risc în exploatarea lacurilor de acumulare [13]

Lucrarea prezintă o abordare *deterministă* a problemelor de exploatare a lacurilor de acumulare cu scopuri multiple, menită să conducă la evitarea unor situaţii extreme, determinînd întreaga mulţime de decizii posibile care garantează funcţionarea eficientă a sistemului. Aceasta permite organului de exploatare atît să aleagă acea decizie care se potriveşte cel mai bine cu informaţiile suplimentare sau cu prognoza de care dispune, cît şi să o adapteze la obiectivele secundare care nu au fost considerate în formularea iniţială a problemei.

Regula de exploatare este de tipul $r_\tau^i = r(\tau, x_\tau^i, a_\tau^i)$, unde x_τ^i reprezintă nivelul (volumul) în lac la începutul zilei τ a anului i şi a_τ^i afluxul în lac.

Regula optimă de exploatare $r(\cdot)$ va fi selectată cu referiri exprese la performanţele pe care le permite în anumite situaţii specifice definite printr-un set $I = \{(a_\tau^i);\ \tau = 1, \ldots, 364,\ i = 1, \ldots, n\}$ — set de referinţă.

Problema se formulează ca o problemă de optimizare cu dublu obiectiv, în care urmează să se minimizeze cel mai mare deficit de apă ($\max D^i$) şi cel mai mare număr de zile cu inundaţii ($\max F^i$)

$$\left\{\begin{matrix} \min \\ X_0,\ r(\cdot) \end{matrix}\right\} \left| \begin{matrix} \max D^i & \max F^i \\ 1 \leqslant i \leqslant n & 1 \leqslant i \leqslant n \end{matrix} \right| \tag{1}$$

X_0 — set al valorilor iniţiale.

Restricţiile problemei sînt:

$x_{\tau+1}^i = x_\tau^i + a_\tau^i - r_\tau^i$ — ecuaţia de continuitate (2)

$0 \leqslant r_\tau^i \leqslant S(x_\tau^i)$ — restricţie fizică (3)

$S(x_\tau^i)$ — funcţia de descărcare a lacului.

$x_\tau^i \geqslant \underline{x}$; $r_\tau^i = S(x_\tau^i)$ dacă $x_\tau^i > \overline{x}$ — restricţie legală (reglementare) (4)

$x_{365}^i = x_0$ — restricţie terminală (5)

Soluţionarea ambelor probleme conduce la reguli de exploatare de forma:

$$r_\tau^i = r(\tau, x_\tau^i, a_\tau^i, D_\tau^i, F_\tau^i, D^*, F^*) \tag{6}$$

în care D* ş F* sînt valorile de referinţă (limite superioare considerate).

Pentru aceste seturi de reguli de exploatare, o procedură simplă va determina pe acelea care garantează valoarea minimă a lu F* pentru orice valoare dată a lui D*.

În general, aceste reguli de exploatare eficientă nu sînt unice. Dată fiind valoarea curentă a informaţiilor (τ, x^i_τ, a^i_τ, D^i_τ, F^i_τ), algoritmul sugerează o gamă întreagă de debite defluente posibile r^i_τ (fig. 4).

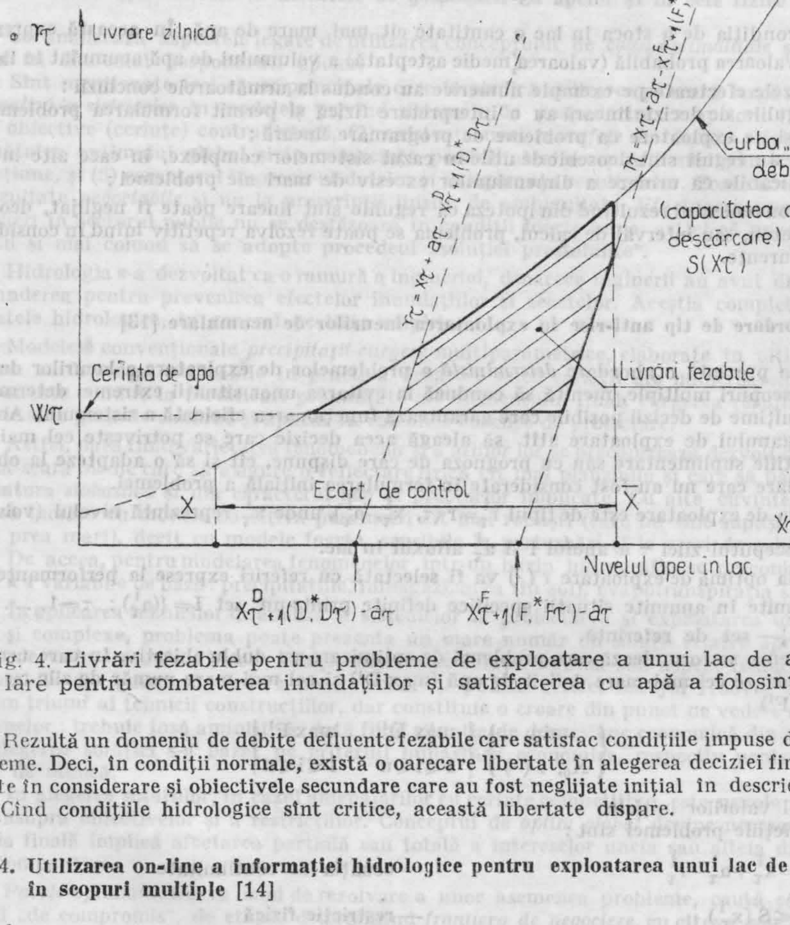


Fig. 4. Livrări fezabile pentru probleme de exploatare a unui lac de acumulare pentru combaterea inundaţiilor şi satisfacerea cu apă a folosinţelor.

Rezultă un domeniu de debite defluente fezabile care satisfac condiţiile impuse de cele două probleme. Deci, în condiţii normale, există o oarecare libertate în alegerea deciziei finale, putînd fi luate în considerare şi obiectivele secundare care au fost neglijate iniţial în descrierea problemei. Cînd condiţiile hidrologice sînt critice, această libertate dispare.

3.4. Utilizarea on-line a informaţiei hidrologice pentru exploatarea unui lac de acumulare în scopuri multiple [14]

În general, preocupările privind optimizarea exploatării lacurilor de acumulare iau în considerare o cunoaştere „perfectă" a elementelor hidrologice de referinţă (debite, precipitaţii etc.) sub formă *deterministă* sau *probabilistă*.

Această ipoteză este nerealistă şi de aceea sînt de subliniat eforturile pentru realizarea de sisteme informaţionale care să asigure, în timp util, colectarea, prelucrarea şi analiza datelor hidrologice. Nu s-a acordat însă suficientă atenţie unor analize vizînd beneficiul obţinut din îmbunătăţirea exploatării lacurilor de acumulare pe baza perfecţionării sistemelor informaţionale hidrologice.

Se prezintă rezultatele privind evaluarea beneficiilor obţinute din sistemele care asigură informaţii hidrologice *on-line* în timpul perioadelor de viitură, în cazul unui lac de acumulare din bazinul rîului Mondego (Portugalia). Evaluarea se referă atît la exploatarea curentă a lacu-

lui de acumulare în timpul producerii viiturilor, cît şi la reglarea adaptivă a volumului de apă din lac pentru atenuarea viiturii. În legătură cu acesta din urmă se preconizează înlocuirea *curbei „ghid superior"* utilizată în graficul de exploatare a acumulării, cu conceptul de *funcţie „ghid superior"*. Folosind conceptul menţionat, informaţia obţinută din prognoza hidrologică permite, pentru un volum de lac pentru atenuare V, diminuarea riscului anual de inundare r ; pentru un r dat este posibilă reducerea substanţială a volumului pentru atenuare V şi deci, creşterea eficienţei în utilizarea lacului pentru celelalte folosinţe.

Exploatarea lacului de acumulare se formulează, din punct de vedere al estimării valorii informaţiilor hidrologice, ca un proces în două faze :

— determinarea volumului de atenuare V, în afara perioadelor de ape mari ;
— controlul debitului defluent în timpul perioadelor de ape mari (viitură).

Analiza efectuată în cazul prezentat s-a axat pe evaluarea efectului informaţiei hidrologice în cazul considerării volumului de atenuare V ca parametru fix şi respectiv în condiţiile reglării adaptive a lui V. În ambele cazuri s-au considerat trei situaţii privind informaţiile hidrologice, şi anume :

— situaţia 1, în care nu se dispune de loc de date hidrologice *on-line* ;
— situaţia 2, în care se dispune parţial de informaţii *on-line* privind afluxul în lac şi pe zona necontrolată aval de lac ;
— situaţia 3, în care se dispune de informaţie hidrologică perfectă *on-line*, sub forma deterministă.

Funcţia „ghid superior" pentru reglarea adaptivă a lui V s-a determinat sub forma unei familii de curbe calculate pe baza afluxului mediu în lac în ultimele 20 zile.

Ca rezultate mai importante ale analizei efectuate se menţionează următoarele :

— riscul de inundare, în fiecare perioadă de ape mari considerată, este redus sensibil prin asigurarea prognozei *on-line* a debitelor de viitură provenite din zona de bazin aval de lacul de acumulare ;

— volumul de atenuare poate fi redus în mare măsură fără a afecta sensibil riscul anual de inundare în zona apărată ;

— prin utilizarea prognozei hidrologice de scurtă durată privind afluxul de apă în lac şi prin reglarea adaptivă a volumului de atenuare se obţine un spor de producţie de energie electrică de 15 GWh, în condiţiile unui an hidrologic mediu.

3.5. Utilizarea informaţiilor cu caracter vag în probleme de alocare a resurselor [15]

Materialul abordează probleme de alocare a resurselor de apă şi alte resurse naturale, care pot fi tratate sub formă de probleme de programare matematică cu parametri *vagi* şi sugerează un mod de abordare pentru acele tipuri de probleme care permit să se determine *alternative nedominate*.

Referindu-se la modelarea matematică a unor sisteme reale, se arată că modelele tradiţionale nu pot cuprinde decît un număr limitat de factori ai sistemelor sau fenomenelor studiate, fapt care poate fi ameliorat prin utilizarea unor parametri neprecizaţi, mai bine zis printr-o modelare care ţine seama de caracterul *vag* al acestor parametri. Se arată că teoria *mulţimilor vagi* (*fuzzy*) aduce elementele necesare în acest sens, cu menţiunea că un sistem real poate fi obiectul unei modelări cu ajutorul unor mulţimi şi al unor parametri *vagi*, cu particularităţile specifice domeniului, deci cu definiţiile şi terminologia adecvată.

Pentru ilustrare se prezintă un exemplu de problemă de programare matematică liniară :

Pentru n tipuri de culturi diferite se caută repartizarea parcelelor de pe o suprafaţă totală A, astfel încit beneficiul total să fie maxim.

Problema, în formulare tradiţională, este exprimată de relaţia :

$$\sum_{j=1}^{n} p_j c_j x_j \rightarrow \max$$

cu condiţiile :

$$\sum_{j=1}^{n} w_j x_j \leqslant W, \quad \sum_{j=1}^{n} x_j \leqslant A, \quad x_j \geqslant 0, \quad (j=1, \ldots, n).$$

unde :

p_j este beneficiul pe unitatea de producţie a culturii j ;
x_j — suprafaţa alocată pentru cultura j ;
c_j — productivitatea culturii j ;
w_j — cantitatea de apă necesară irigării culturii j pe unitatea de suprafaţă ;
W — cantitatea totală de apă disponibilă pentru irigaţii.

În problema enunţată intervin parametrii c_j, w_j şi W, care în mod natural depind de: conţinutul nutritiv al solului, tehnologiile de tratare a solului, activitatea solară etc., factori ce nu sînt întotdeauna neglijabili.

De aceea, un model mai apropiat de realitate ar fi cel în care :

— parametrii respectivi se consideră nu ca numere, ci ca intervale de variaţii ;

— se iau în considerare unele informaţii suplimentare date sub forma unor coeficienţi de pondere.

Se ajunge astfel la o formulare a problemei de programare matematică de tip *fuzzy*, impunîndu-se ca cerinţele din condiţia de mai sus să fie respectate cu valori ale coeficienţilor c_j ($j=1, \ldots, n$) şi w_j date ca mulţimi *vagi*.

Problema se formulează ca o problemă de programare matematică *fuzzy*, cu funcţia obiectiv reprezentată de o aplicaţie *fuzzy*. Se stabileşte apoi o relaţie de preferinţă de tip *fuzzy* între alternativele realizabile (posibile) şi este definită funcţia de apartenenţă a submulţimii de *alternative nedominate*.

Considerînd gradul dorit de *nedominanţă* α şi nivelul de *fezabilizate* β, alternativa care îndeplineşte aceste cerinţe se poate determina rezolvînd problema de programare matematică :

$$\sum_{j=1}^{n} p_j c_j x_j \rightarrow \max_{x_j,\, c_j,\, w_j}$$

cu condiţiile :

$$\sum_{j=1}^{n} w_j x_j \leqslant W, \quad \sum_{j=1}^{n} x_j \leqslant A$$

$$x_j(c_j) \geqslant \alpha \; ; \; v_j(w_j) \geqslant \beta \quad j=1, \ldots, n$$

$$x_i \geqslant 0$$

Prin modificări ale vectorilor α şi β se pot determina alternative cu diferite echilibre între gradul de *nondominanţă* şi cel de *fezabilitate*.

În final se evidenţiază avantajul introducerii unor asemenea raţionamente de tip *fuzzy* pentru apropierea între limbajul folosit în modelele matematice şi limbajul mai puţin precis, mai *vag*, folosit de utilizatorii potenţiali ai acestor modele.

3.6. Sistem de analiză şi fundamentare a deciziilor pentru programele de amenajare în domeniul apelor [16]

Pentru probleme de decizii privind eşalonarea proiectelor, decalarea sau eliminarea din programul de lucrări, se propune o combinaţie de modele matematice, precum şi de cunoştinţe şi intuiţii ale decidentului într-un *sistem suport pentru decizii* „om-calculator".

Sistemul de analiză ş. fundamentare propus este destinat să ajute pe decident în următoarele direcţii :

a) definirea exactă a politicii/strategiei pentru care urmează a se lua decizia şi mai buna înţelegere a organizării ;

b) considerarea majorităţii datelor semnificative care caracterizează proiectul ;

c) calculul metodic al importanţei datelor proiectului supus deciziei, în raport cu politica globală, luînd în considerare parametrii calitativi ;

d) gradarea proiectului în concordanţă cu datele strategice şi cu deciziile anterioare şi alegerea variantei ;

e) examinarea relaţiei dintre strategia preconizată iniţial şi strategia adoptată ;

f) definirea strategiei în luarea de decizii fundamentale; stabilirea nivelului de însemnătate şi definirea detaliată a unei variante se pot face de către tehnicieni de nivel mai puţin ridicat.

Sistemul este alcătuit din: echipamentul de calcul, un set de programe şi proceduri de calcul şi decident. Ca o parte specifică, sistemul include algoritmul Saaty pentru definirea cantitativă a strategiei.

Modelul pentru ierarhizarea preliminară a proiectelor este alcătuit din două submodele:

— *submodelul 1* priveşte proiectele în curs de execuţie şi calculează pagubele care ar decurge din decizia de a opri sau de a amîna execuţia proiectelor respective. Pagubele se calculează în funcţie de o serie de parametri (investiţia neutilizată, ponderea în amenajare, efortul de investiţie imobilizat, prioritatea funcţională, complexitate, prioritate pe termen lung, siguranţa balanţei apei, pagube cauzate subcontractanţilor etc.), ataşîndu-le coeficienţi de pondere relativă definiţi de către decident;

— *submodelul 2* priveşte proiectele care se găsesc într-un stadiu preliminar şi calculează *utilitatea* anticipată, ca un număr cu valoare între 0 şi 1, în funcţie de aportul lor sub aspect tehnic, de economicitate, nivelul de risc şi de considerente generale. Parametrii cu diferite dimensiuni se *normalizează* printr-o relaţie de conversie, pentru comparabilitate.

După ierarhizarea preliminară, valoarea *utilităţii* fiecărui proiect se corectează, în cursul desfăşurării procesului de decizie, pe măsura obţinerii de precizări privind parametrii luaţi în considerare. În acest scop se folosesc probabilitatea *a priori*, precum şi probabilitatea *a posteriori*, calculată cu ajutorul teoremei lui Bayes.

Pentru ca decidenţii să poată „dialoga" direct cu calculatorul, s-au realizat dispozitive de afişaj cu posibilitate de reglare a regimului de afişare, în funcţie de experienţa utilizatorului.

3.7. Descompunerea, coordonarea şi agregarea în conducerea optimală a unei mari reţele de alimentare cu apă [17]

Se prezintă problema de conducere optimală, în timp real, a unei reţele relativ mari de alimentare cu apă, care include peste 20 de rezervoare. Prin exploatarea particularităţilor acestui sistem, format din subreţele interconectate cu interconexiuni controlabile, este posibilă cooperarea acestora într-o structură ierarhică complexă pentru a realiza optimizare şi reglare directă. Deşi în prezent se asigură urmărirea funcţionării printr-un sistem de teletransmisie, reglajul trebuie considerat manual, deoarece actul decizional se sprijină doar pe experienţa operatorului.

În 1974, societatea SLEE (Société Lyonnaise des Eaux et de l'Éclairage) a decis să cerceteze posibilităţile oferite de teoria optimizării în vederea producerii unui pachet de programe care să permită luarea sistematică de decizii cu ajutorul calculatorului. Studii preliminare efectuate pe o subreţea de 3 rezervoare principale au dus la concluzii pozitive în 1977. Din punct de vedere matematic, s-au utilizat în principal idei de centralizare/descentralizare, care s-au transformat într-o procedură euristică iterativă, obţinîndu-se rezultate satisfăcătoare (comparativ cu comportamentul simulat al operatorului uman).

Noul studiu comentat în articol a început în 1980 şi intenţiona să producă în final un pachet de programe operaţional. În acest scop o firmă particulară de software (SINAC) a fost asociată cu proiectul. Software-ul care a rezultat este foarte modular şi se poate adopta multor alte cazuri, deoarece reţeaua considerată oferă majoritatea dificultăţilor care se pot întîlni la probleme de acest nivel.

Problema a fost împărţită în subprobleme mai mici, fără a renunţa la optimizarea globală, care depinde de asemenea de exploatarea optimă a interconexiunilor (pe lîngă controlul optim al fiecărei subreţele). În ciuda împărţirii, unele subprobleme rămîn complexe şi scumpe pentru a fi rezolvate de mai multe ori, aşa cum cere procesul de coordonare. S-a făcut deci apel la idei de centralizare pentru a simplifica rezolvarea acestor subprobleme.

Întregul procedeu de optimizare *indirectă* este efectuat în timpul nopţii, pe un orizont de o zi, bazat fiind pe consumul de apă preliminat. Deoarece consumul real, evident, va diferi de valorile prevăzute, şi pentru că există şi alte inexactităţi ale modelului, este necesară de asemenea computerizarea *directă*. O parte din pachetul de programe pentru optimizarea *indirectă* poate fi utilizată direct pentru a compensa aceste abateri, dar cu un orizont limitat de unul sau de cîteva intervale de timp, evitîndu-se astfel partea cea mai scumpă a computerizării, adică optimizarea *dinamică*.

Simulări de cîteva zile s-au efectuat utilizîndu-se atît strategia computerizată prin program, cît şi cea folosită de operator pentru zilele corespunzătoare. Toate rezultatele au fost în favoarea programului. Mai mult, strategia generată de program a dus la o bună comportare calitativă.

Deşi nu se aşteaptă mari economii pe termen scurt, principalele avantaje pe care le oferă software-ul sînt cele ale oricărui automat : performanţe uniforme şi reproductibile, adaptare imediată la orice schimbare provizorie sau definitivă în reţea şi de asemenea o mai bună înţelegere a limitărilor sistemului şi deci un ajutor pentru planificarea viitoare, posibilitatea de a evalua mai raţional viitoarele investiţii etc.

Pachetul de programe este planificat să devină operaţional la începutul anului 1984. Sistemul se va utiliza ca un sistem de luare a deciziilor cu ajutorul calculatorului şi va oferi următoarele date operatorului :

— optimizare *indirectă*, cu cererea prevăzută ;

— optimizare *directă* de „închidere a golurilor" (în fiecare oră) ;

— simulare directă a viitoarelor funcţionări ale reţelei la cererea operatorului, cu strategii de reglare optimizate sau cu date *a priori* ;

— înmagazinare de date şi evaluare a consumurilor reale ;

— evaluarea *a posteriori* a performanţelor indirecte ale strategiilor operatorului şi calculatorului faţă de consumul real etc.

Pe lîngă utilizarea curentă zilnică, softul realizat va putea fi utilizat în optimizări şi simulări ale funcţionării reţelei de alimentare cu apă în scopuri de planificare.

3.8. Optimizarea pompării în sistemele de alimentare cu apă [18]

Sistemele de alimentare cu apă tipice cuprind un mare număr de conducte interconectate, precum şi ventile şi pompe. Reţelele sînt prevăzute deseori cu capacitate de stocare sub forma unor rezervoare de serviciu. Date obţinute din practică arată că o proporţie semnificativă din cheltuielile totale de exploatare se datorește costului energiei electrice pentru pompare. Aceasta furnizează un motiv puternic pentru optimizarea operaţiunilor de pompare. Local, obiectivul optimizării este de a selecta o combinare a pompării în scopul minimizării cererii de *putere* necesară. Global, obiectivul principal este de a determina strategii ale pompării pentru minimizarea costurilor cu energia de pompare.

De aceea, în condiţii specifice, problema generală a pompării optimizate în sistemele de alimentare cu apă se poate împărţi în două direcţii principale : utilizarea optimizată a pompării şi programarea optimizată a pompării.

Utilizarea optimizată a pompării este o problemă de optimizare *statică*, în care se caută un regim adecvat al pompării, cu consum minim de putere, pentru a satisface condiţiile instantanee de exploatare a sistemului.

Programarea optimizată a pompării este o problemă de optimizare *dinamică*, în care se ţine seama de stocajul de apă din sistem şi de variaţia în timp a tarifelor de electricitate, în scopul minimizării costurilor totale de exploatare într-o anumită perioadă de analiză. Ambele aspecte trebuie avute în vedere pentru studiul optimizării sistemelor de alimentare cu apă.

Determinarea unor scheme de acţiune îmbunătăţite necesită modele corecte ale reţelelor de distribuţie. Totuşi, interacţiunea puternică între componentele neliniare ale reţelei şi numărul mare de variabile ale sistemelor impun severe restricţii teoretice şi de computerizare în aplicarea tehnicilor de optimizare pentru analiza funcţionării sistemelor. S-au făcut mai multe propuneri de optimizare a anumitor sisteme. În cele mai multe cazuri, s-au făcut presupuneri restrictive, care ori simplifică consideraţiile de optimizare reţinînd modelul general al sistemului, ori încearcă optimizarea completă dar cu o reprezentare simplificată a sistemelor.

Staţiile de pompare au, de obicei, combinaţii în paralel de pompe cu caracteristici diferite. Aceasta permite o reglare de tip *discret* sau o reglare de tip *mixt* cu variabile atît *discrete*, cît şi *continue*. Obiectivul primordial în optimizarea pompării este de a determina combinaţii potrivite între aceste variabile. Formularea care rezultă formează baza de dezvoltare a procedeelor de programare optimizată ; acestea sînt utilizate împreună cu tehnica de programare dinamică.

Programarea dinamică este utilizată ca procedeu de bază în optimizarea planificării pompării, cu selectarea combinaţiilor de pompare bazate pe consideraţii de folosire a pompelor. Soluţia defineşte acţiunile necesare atît *discrete*, cît şi *continue*, corespunzînd combinaţiilor de funcţionare-nefuncţionare ale pompelor şi vitezelor de pompare.

Pentru calculul pompării optime momentane şi a programului optim de pompare a fost elaborat un program de calculator pentru lucru *interactiv*, cu ilustrări grafice, program aplicat din anul 1982 la un sistem existent de alimentare cu apă.

Rezultatele obţinute privesc în special :
— creşterea pompării în timpul nopţii ;
— utilizarea celor mai eficiente pompe ;
— reducerea comutărilor de pe o pompă pe alta ;
— limitări ale variaţiei nivelelor din rezervoare ;
— limitări ale puterii maxime cerute ;
— diminuarea debitelor necesare la sursa de alimentare.

Se pot obţine astfel reduceri ale costului de ordinul 5—10 %, reduceri care sînt semnificative avîndu-se în vedere că schemele tipice de alimentare cu apă din Anglia au costuri de exploatare anuale care depăşesc 1 milion lire sterline.

BIBLIOGRAFIE

1. *Y. Nakamori, Y. Sawaragi* (Japonia) **Surveillance network design for air pollution control** (Proiectarea reţelelor de supraveghere pentru controlul poluării aerului). Indicativ **11.7-2.**
2. *R.P. Ibbitt, P.D. Hutchinson* (Noua Zeelandă) **Model parameter consistency and fitting criteria** (Consistenţa parametrilor modelelor matematice în hidrologie — şi criterii de calibrare). Indicativ 11.8/B-3.
3. *A. Szöllösi-Nagy*, ş.a. (R.P.U.) **On the use of recursive algorithms in real-time river flow forecasting — Hungarian experiences** (Asupra folosirii algoritmilor recursivi în prognoza hidrologică în timp real — Experienţa ungară). Indicativ 11.8/C-5.
4. *A. Esogbue* (S.U.A.) **Using fuzzy sets and hierarchical models in non point source water quality management** (Folosirea teoriei mulţimilor vagi şi a modelelor ierarhizate la protecţia calităţii apelor în cazul unor impurificări neconcentrate). Indicativ 11.8/A-6.
5. *M. Cremer, J. Kappenberg* (R.F.G.) **Surveilling water pollution in a tidal river by a state estimator with a switching structure** (Supravegherea poluării apei într-un rîu influenţat de fenomenele de maree, folosind un estimator de stare cu structură comutabilă). Indicativ 11.7-3.
6. *M. Papageorgiou* (R.F.G.) **Multilayer control of sewer networks** (Conducerea multinivel a exploatării reţelelor de canalizare). Indicativ 11.7-4.
7. *M. Hiraoka, K. Tsumura* (Japonia) **Modelling and control of the activated sludge process by use of autoregressive model** (Modelarea şi controlul procesului cu nămol activ utilizînd modelul autoregresiv). Indicativ 11.8/C-1.
8. *H. Kurz* (R.F.G.) **Adaptive control of a waste water neutralization process** (Reglarea adaptivă a unui proces de neutralizare a apelor uzate). Indicativ 11.7-6.
9. *K. Maeda* (Japonia) **A knowledge based system for the wastewater treatment process** (Sistem bazat pe cunoaştere pentru conducerea procesului de epurare a apelor uzate). Indicativ 11.7-5.
10. *M.B. Beck* **Operational control and estimation in Water Quality Management** (Reglare şi evaluare operativă în gospodărirea/protecţia calităţii apelor).
11. *Myron B. Fiering* (S.U.A.) **Near optimality in management and physical models** (Cvasi-optimalitate în modelele de gospodărire a apelor şi în cele fizice). Indicativ 11.8/B-1.
12. *H. Pietkiewicz-Saldan* (Polonia) **Analysis and physical interpretation of decision rules of operative water system management** (Analiza şi interpretarea fizică a regulilor de decizie privind regimul de exploatare a unui sistem de gospodărire a apelor). Indicativ 11.8/C-6.
13. *G. Guariso* (It.), *S. Orlovski* (Austria), *S. Rinaldi* (It.) **A risk-averse approach for reservoir management** (Abordare de tip anti-risc în exploatarea lacurilor de acumulare). Indicativ 11.8/C-3.
14. *L. Valadares Tavares* (Portugalia) **Value of on-line hydrologic information for the operation of a multipurpose reservoir** (Utilizarea „on-line" a informaţiei hidrologice pentru exploatarea unui lac de acumulare în scopuri multiple). Indicativ 11.8/B-5.

15. *S.A. Orlovski* (Austria) **Fuzzy information in problems of resources allocation** (Informaţii cu caracter vag în probleme de alocare a resurselor). Indicativ 11.8/A-1.

16. *Y. Carmon, M. Ben-Bassat* (Israel) **A decision support system for large scale water development programs** (Sistem suport pentru decizii în programele de amenajare în domeniul apelor). Indicativ 11.8/C-2.

17. *P. Carpentier, C. Cohen* (Franţa) **Decomposition, coordination and aggregation in the optimal control of a large water supply network** (Descompunerea, coordonarea şi agregarea în conducerea optimală a unei mari reţele de alimentare cu apă). Indicativ 11.8/C-4.

18. *B. Coulbeck, C.H. Orr* (Anglia) **Optimized pumping in water supply systems** (Pompare optimizată în sistemele de alimentare cu apă). Indicativ 11.8/B-4.

APLICAŢII SPAŢIALE

Ing. *M. Sîrbu*
I.P.A.

1. INTRODUCERE

Lucrările *secţiunii 12.0* se ocupă cu aplicaţii în domeniul astronauticii. Cunoaşterea universului a reprezentat din totdeauna un domeniu de mare atracţie pentru oamenii de ştiinţă, însă numai revoluţia tehnologică actuală a făcut posibilă studierea eficientă a acestuia. Ceea ce putem cunoaşte despre univers depinde de mijloacele de observare şi de metodele de măsurare pe care actuala dezvoltare tehnică ni le pune la dispoziţie.

Limitele universului observabil se situează astăzi în jurul a 15 miliarde de ani lumină, ceea ce înseamnă că dacă am face o hartă a universului observabil pe care Luna s-ar găsi la 1 mm de Terra, această hartă ar acoperi o suprafaţă de 500 miliarde kilometri pătraţi.

Cîteva din lucrările prezentate în cadrul secţiunii 12.0 analizează soluţiile la care s-a recurs pentru efectuarea observaţiilor astronomice. Alte lucrări propun metode noi, mai eficiente, de investigare a spaţiului cosmic. O problemă importantă abordată de lucrările acestei secţiuni se referă la poziţionarea navelor cosmice, prezentîndu-se sistemele de reglare automată utilizate în acest scop, analizîndu-se măsura în care acestea satisfac necesităţile ridicate de cercetarea ştiinţifică a spaţiului cosmic.

2. REGLAREA POZIŢIEI NAVELOR SPAŢIALE

Telescopul spaţial descris în *lucrarea 12.0/A-1* este cel mai mare şi mai puternic observator optic astronomic în ultraviolet care va funcţiona în spaţiu şi care va permite obţinerea unor imagini din spaţiul cosmic de o calitate inegalabilă. Distanţa nominală a orbitei pe care se situează telescopul pentru efectuarea observaţiilor asupra spaţiului cosmic este de 550 km. În lucrare sînt descrise şi instrumentele complementare asociate observatorului, sistemul optic şi sistemul de control al vizării.

Spre deosebire de telescoapele relativ mici care l-au precedat, telescopul spaţial al NASA are gabaritul şi durata de viaţă mult mai mari. Lansarea în spaţiu, desfăşurarea, deservirea şi readucerea pe Pămînt a telescopului se vor realiza cu ajutorul sistemului de transport al navetei spaţiale. După lansare se manevrează pentru atingerea orbitei şi poziţiei corespunzătoare, după care urmează separarea telescopului de sistemul de transport al navetei spaţiale. În timpul conectării telescopului la sistemul de separare, sau imediat după lansarea sa în spaţiu, are loc desfăşurarea antenei de înaltă frecvenţă şi a aripilor solare şi totodată sînt iniţiate testările funcţionale de sistem în scopul de a verifica dacă observatorul funcţionează corespunzător. Din acest moment începe misiunea telescopului, iar sistemul de transport al navetei spaţiale este readus pe Pămînt. Deoarece treptat telescopul părăseşte orbita stabilită, este necesară executarea unor manevre de menţinere pe orbită. Acestea vor fi efectuate de către Orbiter în punctul de întîlnire. Este posibilă aducerea observatorului pe Pămînt pentru realizarea unor verificări amănunţite.

Telescopul spaţial este o structură cilindrică cu lungimea de 13,1 metri, cu diametrul de 4,27 metri şi cu greutatea de 11 000 kilograme. Elementele de bază din configuraţia telescopului spaţial sînt ansamblul telescopic optic (ATO), un modul de protecţie a sistemului (MPS), instrumentele ştiinţifice (IS) şi aripile solare (AS). Cele două aripi solare pot fi desfăşurate sau retractate la comandă şi pot fi rotite în jurul axelor transversale ale observatorului pentru menţinerea poziţiei perpendiculare pe linia soarelui. ATO este format dintr-un telescop de tip Cassegrain, care este plasat în interiorul unui scut meteoric şi al unei apărătoare contra soarelui. MPS este modulul de protecţie, care cuprinde în interiorul său ATO şi IS, asigurînd în acelaşi timp legătura între Orbiter şi operaţiile executate la sol. MPS furnizează elementele de bază de care depinde funcţionarea corectă şi precisă a observatorului. Instrumentele ştiinţifice sînt

plasate în planul focal, în spatele oglinzii primare și sînt menținute în poziție fixă. Telescopul are 5 instrumente științifice și un instrument pseudoștiințific, fiecare dintre acestea fiind un modul separat care poate fi deplasat în spațiu.

O problemă importantă abordată în cadrul lucrării se referă la reglarea vizării și stabilității. Elementele care condiționează proiectarea sistemului de reglare a stabilității depind de calitatea sistemului optic al telescopului spațial și de sensibilitatea instrumentelor științifice asociate telescopului. În cadrul metodei de proiectare pentru care s-a optat există două posibilități de reglare a poziției de funcționare : reglarea poziției în raport cu soarele (R.P.S.) și reglarea operațională a vehiculului (ROV). RPS este utilizată în timpul desfășurării inițiale a observatorului, cînd au loc testările de echipament înainte de misiune, precum și în timpul operațiilor neprevăzute. Modulul de reglare ROV asigură orientarea și menținerea axelor optice ale telescopului spre coordonate de interes științific, pentru efectuarea unor observații ce pot dura de la cîteva minute pînă la cîteva zile.

Telescopul spațial va fi un instrument puternic pe durata celor 50 sau mai mulți ani de viață și va fi accesibil oricărui cercetător științific calificat. Operațiile efectuate cu acest telescop vor fi în cea mai mare parte asemănătoare cu cele efectuate de alte centre astronomice cu baza la sol, exceptînd acele operații care se referă la conducerea unui telescop situat la distanță mare în spațiul cosmic.

Cercetările nu au fost încă definitivate, astfel încît, ținînd seama de structura sistemului telescopului spațial și de problemele ridicate de funcționalitatea acestuia, se impun atenției problemele ridicate de lucrarea *12.0/A-5*, în ceea ce privește panourile solare și problemele ridicate de lucrările *12.0/A-2*, *12.0/A-3*, *12.0/A-4*, *12.0/A-6* în legătură cu reglarea poziției. Deoarece se preconizează ca pe durata celor 50 de ani de viață telescopul să nu fie adus la sol, sau să fie adus foarte rar, o problemă importantă care se ridică este aceea de a asigura energia necesară pentru funcționarea întregului sistem pe această durată. Aripile cu baterii solare pot constitui o sursă de energie convenabilă.

Controlul flexibilității aripilor cu baterii solare constituie tema *lucrării 12.0/A-5*. Rolul acestor aripi este de a colecta în mod eficient energie solară prin controlul precis al orientării către soare. Limitările impuse în legătură cu greutatea rachetelor lansate în spațiu impun ca aripile să fie construite din materiale ușoare și foarte flexibile. Sistemul format de aceste aripi flexibile va fi un sistem cu parametri distribuți, deci va avea mai multe moduri de vibrație. Aripile sînt modelate ca un sistem în consolă flexibilă, vibrînd numai prin încovoiere, neglijîndu-se vibrațiile prin torsiune. Soluțiile formulării matematice nu sînt prezentate în această lucrare datorită spațiului limitat. Dinamica sistemului este analizată pe baza modelului în consolă al aripilor cu baterii solare.

În prezent se lucrează la o metodă de reglare pe baza estimării modurilor de vibrație ale aripilor. Există două tehnici de estimare corespunzătoare celor două tipuri de senzori care pot fi utilizați în spațiu : aparatul de măsurare a deformării la încovoiere și celulele instrumentului cu baterii solare. Cei doi senzori pot măsura momentele de încovoiere și respectiv unghiurile de deviație. Metoda de estimare propusă este cea corespunzătoare celui de al doilea tip de senzor, aceasta fiind mai adecvată condițiilor din spațiu.

O altă problemă tratată în lucrare se referă la reglarea maximală a ieșirii. Aceasta este una dintre cele mai importante probleme, deoarece este legată de orientarea precisă a aripilor în funcție de soare pentru a maximiza cantitatea de lumină ce cade pe aripă. Spre deosebire de metoda de reglare convențională, metoda de reglare modificată propusă în lucrare ia în considerare flexibilitatea aripii.

Fie S_i și S_{i+1} ieșirile aripii cu baterii solare la momentul i și respectiv după efectuarea unei mici rotații către soare. Schema de reglare statică convențională va fi :

(1) Dacă $Si \geqslant Si+1$, atunci aripa se rotește în continuare în aceeași direcție, cu un unghi prescris, pentru a mări cantitatea de lumină ce cade pe aripă ;
(2) În caz contrar, rotația se face în direcția opusă ;
(3) Se continuă rotația în sensul pentru care s-a luat decizia mai sus, pînă cînd ieșirea devine maximă.

Metoda modificată de reglare pornește de la estimarea, pe baza datelor obținute prin măsurarea cantității de lumină ce cade pe aripa nedeviată. Cu această estimație, reglarea se implementează utilizînd exact același algoritm convențional descris mai sus.

O altă problemă importantă este ridicată de necesitatea de a asigura un sistem de reglare tolerant la defecte, care este absolut necesar atunci cînd se lucrează în spațiu. Se va utiliza conceptul de reglare duală cu detectarea defectului și diagnosticare. Dacă un senzor se defectează, defectul poate fi detectat și izolat în cadrul algoritmului de reglare, evitîndu-se în acest

fel ca reglarea să se facă pe baza unor informaţii greşite. Detectarea defectului şi diagnosticarea sînt realizate cu un sistem de reglare biprocesor. Experimentele făcute au dat rezultate bune. La apariţia unui defect sistemul de reglare va lucra în mod recuperativ, adică reglarea se modifică astfel încît să anuleze efectul produs de apariţia defecţiunii.

În *lucrarea 12.0/A-5* s-a pus în evidenţă necesitatea de a lua în consideraţie elasticitatea elementelor constructive ale navei spaţiale. Această problemă este abordată şi în cadrul *lucrării 12.0/A-6*, aici însă centrul de interes cade pe proiectarea regulatorului care controlează poziţia navei spaţiale cu structură elastică. Proiectarea se realizează pe baza metodelor frecvenţiale. Se începe cu descrierea sistemului prin ecuaţii diferenţiale parţiale cu model de amortizare intern. Structura elastică este constituită din doi arbori elastici ataşaţi de o parte şi de alta a unui disc rigid. Funcţia de transfer care descrie comportarea acestei structuri este următoarea :

$$\theta_1=[1/s^2\,(I_1+I_2+I_3)]\,G_m\,(s)-T_1 \tag{1}$$

unde :

$$G_m\,(s)=\left[1+\sum_{i=1}^{3}\delta_i\right]\Bigg/\left[1+\sum_{i=2}^{3}\delta_i\,(+hY_is)/Y_is)/Y_is\right]$$

este funcţia de transfer a sistemului mecano-elastic,

$Y_i=\sqrt{I_i\,[K_i+sd_i\,(s)]}$ şi $\delta_i=I_i/I_1$ — raportul de inerţie ;

I_1 — mometul de inerţie al discului rigid [kgm²] ;

I_i — momentul de inerţie total al fiecărui arbore [kgm²], i=2,3 ;

T_1 — torsiunea de reglare exercitată asupra discului [N] ;

θ_i — unghiul de rotaţie al arborelui flexibil.

Acest model poate fi adus la următoarea formă mai simplă :

$$s^2I_1\left[1+\sum_{i=2}^{3}\delta_i\,(\mathrm{th}Y_is)/Y_is\right]\theta_1=T_1. \tag{2}$$

În lucrare se face analiza răspunsului frecvenţial şi se definesc cele trei caracteristici de bază ale sistemului elastic, şi anume : funcţia medie, marginea superioară şi marginea inferioară.

Pentru determinarea funcţiei medii se pleacă de la termenul tangentei hiperbolice G_i (s). Avem :

$$G_i\,(s)=\mathrm{th}\,(Y_is)/(Y_is) \tag{3}$$

unde :

$$Y_i=\sqrt{I_i\,[k_i+s\delta_i\,(s)]}=(1/\omega_i)\sqrt{1+\{\zeta_i\,(s)/\omega_i\}\,s}$$

Caracteristicile frecvenţiale ale termenului tangentei hiperbolice vor fi :

$$\omega_1=\sqrt{k_i/I_i}\,;\quad \zeta_i=\{d_i\,(s)/2\}\sqrt{k_iI_i}$$

unde d_i (s) reprezintă constanta de amortizare

G_i (s) se poate aproxima printr-o funcţie mai simplă astfel :

$G_i\,(s)=\lim_{I_1\to 0} 1/G_m\,(s)$ pentru un singur arbore

G_i (s) este reciproca funcţiei de transfer a sistemului mecano-elastic pentru cazul în care discul este mic şi se ataşează un singur arbore la disc. Structura elastică va fi descrisă prin combinarea a cîtorva sisteme cu parametri distribuiţi de tipul lui G_i (s). Se obţine funcţia de transfer a structurii elastice care este exprimată prin intermediul funcţiei medii, marginii superioare U_a („upper bound") şi marginii inferioare L_a („lower bound"). Proiectarea regulatorului se va face pe baza acestor caracteristici generale. Înainte de a începe proiectarea, structura elastică ce trebuie reglată va fi analizată din 4 puncte de vedere şi anume :

(1) dacă co-locaţia este sau nu satisfăcută ;

(2) dacă frecvenţa de rezonanţă este suficient de mare, este de acelaşi ordin, sau este de ordin mai mic faţă de lăţimea de bandă a sistemului de reglare ;

(3) dacă raportul de inerţie este mare sau mic ;

(4) dacă reglarea arborelui este sau nu necesară.

Există trei categorii de regulatoare :

— cu compensarea amplitudinii ;

— cu compensarea fazei în cadrul lățimii de bandă a elementului de execuție și compensarea amplitudinii în afara acesteia ;
— cu dublă compensare a avansului de fază.

În lucrare se indică situații în care se recurge la una sau alta dintre tipurile de reglare menționate mai sus, această alegere fiind făcută în funcție de rezultatele analizei structurii elastice ce urmează să fie reglată. Se constată că sarcinile de reglare a structurii elastice se împart astfel : sarcini rezolvate de reglarea activă, care acoperă lățimea de bandă a elementului de execuție și sarcini rezolvate de reglarea pasivă, care acoperă domeniul situat în afara lățimii de bandă a elementului de execuție. Din acest motiv caracteristicile de amortizare ale materialelor vor avea aceeași importanță ca și regulatorul și vor fi tratate ca parte a regulatorului.

Această metodă de proiectare, bazată pe caracteristicile generale ale structurii elastice este foarte robustă și are avantajul insensibilității la variații ale parametrilor. Exemplul de proiectare prezentat în lucrare pune în evidență că în cazul oscilațiilor datorate torsiunii, compensatorul optim va cuprinde o buclă de viteză și un filtru trece-jos.

Se preconizează ca, pe viitor, legătura dintre proiectantul structurii și proiectantul sistemului de reglare să se realizeze prin intermediul funcției medii G_t (s) și a caracteristicilor de amortizare ζ_i (s) ale matricelor.

Probleme legate de reglarea poziției mai sînt tratate în cadrul *lucrărilor 12.0/A-2, 12.0/A-3 12.0/A-4*. Acest grup de lucrări se ocupă însă cu reglarea poziției unor sateliți, spre deosebire de celelte lucrări care aveau în vedere aceeași problemă, însă pentru cazul navelor spațiale. *Lucrarea 12.0/A-2* propune o metodă de reglare a poziției unui satelit astrometric cu rotire stabilizată, pe baza schemei cu cuplu de torsiune magnetic, iar *lucrarea 12.0/A-3* propune reglarea numerică în buclă închisă a poziției utilizînd jeturi de gaze.

Legea de reglare a poziției pe baza cuplului de torsiune magnetic propusă în *lucrarea 12.0/A-2* se bazează pe o reacție lineară după abaterea de poziție și asigură atît reglarea vitezei unghiulare cît și amortizarea nutației. Lucrarea pune în evidență posibilitatea de a utiliza în mod avantajos sistemul pe bază de momente de polarizare în combinație cu cuplu de torsiune magnetic pentru sateliții a căror poziție este reglată cu precizie mare în raport cu un sistem de referință inerțial. Se mai propune o nouă schemă pentru manevrele cu unghiuri mari, schemă care se bazează de asemenea pe principiul cuplului de torsiune magnetic. Această schemă încearcă să realizeze o deplasare cvasistatică a referinței de poziție astfel încît să determine o schimbare a poziției în jurul axei de rotație. Se presupune că senzorii de poziție la bord sînt giroscoape integratoare de viteză și magnetometre pe 3 axe. Se mai presupune că la bord există echipament cu microprocesoare. Este important de subliniat că schema pentru manevrele cu unghiuri mari se caracterizează prin simplitatea hardware-lui la bord și a implementării software, cantitatea de prelucrări pe care trebuie să le realizeze calculatorul de la bord fiind mică. Analiza stabilității legii de reglare a poziționării se face prin metoda locului rădăcinilor.

S-au realizat simulări pe calculator pentru cazul satelitului ASTRO-C, care urmează să fie lansat în februarie 1987. Rezultatele pun în evidență faptul că stabilitatea sistemului este asigurată și că precizia de poziționare se poate obține cu ajutorul unui echipament relativ simplu la bord. Rezultatele simulării indică de asemenea că schema propusă pentru manevrele cu unghiuri mari funcționează satisfăcător și că este corespunzătoare pentru aplicarea în cazul sateliților cu rotire stabilizată.

Lucrarea 12.0/A-3 propune o metodă de reglare numerică în buclă închisă a poziției unui satelit astrometric de tip Hipparcos, acționarea fiind asigurată pe bază de jeturi de gaz.

Satelitul Hipparcos, care urma să fie lansat pe o orbită geostaționară la sfîrșitul anilor 1980, este destinat să observe 100 009 de stele cu ajutorul a două telescoape PT și FT, care utilizează în comun același plan focal. Scopul misiunii Hipparcos este acela de a construi un vast catalog al corpurilor cerești, cu o limită de precizie de 0,002″ secunde de arc ; (1 arc sec= =1°/3 600).

Înainte de abordarea principalelor aspecte ale problemei de reglare, în lucrare se prezintă cîteva generalități în legătură cu structura și mișcarea satelitului Hipparcos, și anume se face o scurtă prezentare a sistemului optic, a modului în care decurge observarea corpurilor cerești și mișcarea nominală a satelitului.

Problemele ridicate de reglarea poziției se referă la estimarea poziției, alegerea elementelor de execuție și la legea de reglare în buclă închisă. Procedura de observare în timp real a imaginilor corpurilor cerești impune ca estimarea poziției să se facă la bord, în mod continuu,

cu o precizie de 1 arc sec. Pentru această problemă, un grup de autori (Carllucci şi Donati) propune un algoritm care satisface în acelaşi timp şi cerinţele legii de reglare. La alegerea elementelor de execuţie trebuie să se ţină seama de faptul că nu se admit surse de vibraţie care să producă perturbaţii de poziţie persistente cu o amplitudine mai mare de cîteva miliarcsecunde. Mai mult, este necesar ca evoluţia poziţiei să poată fi modelată cu o precizie de cîteva miliarcsecunde cu ajutorul unui model matematic caracterizat de cei mai mici parametri. În aceste condiţii, cea mai convenabilă soluţie de element de execuţie este cea bazată pe jeturi de gaz, deoarece sînt eliminate vibraţiile care apar între două activări consecutive ale limitatorului de cursă.

La alegerea legii de reglare trebuie să se ţină cont de următoarele cerinţe :

(i) activarea limitatoarelor de cursă să fie simultană pentru axele controlate ;

(ii) legea de reglare operează cu interval de timp T constant între două manevre ;

(iii) numărul N de manevre de reglare să fie cît mai mic pe durata unei perioade corespunzătoare parcurgerii unei orbite.

Aceste cerinţe impun alegerea unei structuri de reglare numerică în buclă închisă cu perioada de eşantionare T fixată şi cît mai mare posibilă. Scopul sistemului de reglare a poziţiei este acela de a menţine eroarea de poziţionare a axei Z şi eroarea mişcării de spin în limitele de toleranţă prescise în condiţiile în care asupra satelitului acţionează perturbaţii necunoscute datorate momentului de torsiune, despre care se ştie că au un caracter periodic. Principala problemă care apare în proiectare constă în evitarea efectului erorilor care apar datorită eşantionării cu T mare a răspunsului sistemului la aplicarea unei reglări cu momente de torsiune în impulsuri. Soluţia originală pe care o dau autorii pentru rezolvarea acestei probleme constă în însumarea semnalelor de referinţă pentru poziţie cu efectele calculate asupra poziţiei satelitului, efecte care apar datorită unor componente de înaltă frecvenţă ale impulsurilor de reglare. Amplificarea pe reacţie datorată acestor operaţii va fi luată în considerare la proiectarea legii de reglare generală.

Se impune ca legea de reglare aleasă, aplicată în prezenţa unor perturbaţii periodice necunoscute, să asigure următoarele performanţe :

— eroarea absolută de poziţionare a axei Z să fie mai mică de 10′ (minute de arc) ;

— eroarea vitezei unghiulare de spin să fie mai mică de 2% ;

— eroarea unghiulară a axelor de spin, în raport cu mişcarea nominală, să fie mai mică de 10′ ;

— să avem valoare mare a lui T într-un domeniu corespunzător lui N=8, 10, 12, în condiţii de operare normale.

Perturbaţiile datorate momentului de torsiune, care apar în condiţii normale de funcţionare, apar datorită presiunii solare şi giroscoapelor.

Admiţînd ipoteza simetriei cilindrice în raport cu axa Z şi a unor perturbaţii mici pe timpul mişcării nominale, ecuaţiile dinamice ale satelitului se descompun după următoarele mişcări independente :

— rotaţia de spin Ψ în jurul axei Z

— interacţiunea mutuală φ şi rotaţiile θ în jurul axelor X, respectiv Y.

S-a proiectat cîte o lege de reglare pentru fiecare dintre aceste mişcări. Reglarea lui Ψ este mai simplă decît reglarea unghiurilor (φ, θ), deoarece în primul caz sistemul dinamic este de ordin mai mic, iar perturbaţiile datorate momentului de torsiune au un număr mai mic de armonice.

În continuare se face o prezentare comparativă a performanţelor în buclă închisă evaluate prin simulare pe calculator, cu performanţele legii de reglare în buclă deschisă, care au fost deduse admiţînd ipoteza că perturbaţiile sînt perfect cunoscute. Rezultatele obţinute, precum şi faptul că reglarea optimală în buclă deschisă se sprijină pe o ipoteză care în condiţiile date nu este valabilă, pun în evidenţă faptul că metoda de proiectare propusă în lucrare este cea mai bună metodă aplicabilă în practică.

Un obiectiv similar cu cel al lucrării 12.0/A-3 are *lucrarea 12.0/A-4*, care preconizează întocmirea unui catalog stelar de mare precizie al întregului cer. Lucrarea propune observarea timp de 2,5 ani a 100 000 de corpuri cereşti, cu ajutorul unui telescop special, montat pe un satelit aflat pe o orbită staţionară în jurul Pămîntului. Datele furnizate de telescop vor fi prelucrate la sol.

Scopul misiunii Hipparcos este de a determina parametrii astrometrici : poziţii (longitudine şi latitudine), mişcări particulare (în longitudine şi latitudine) şi paralaxă trigonometrică pentru fiecare corp ceresc observat.

Observaţiile asupra stelelor vor fi efectuate cu ajutorul unui instrument optic special, format din două telescoape care fixează cerul ca o pereche de compase, şi care au un singur plan focal şi un tub proiector de imagine. Datele colectate de acest instrument vor fi transmise la sol, iar prelucrarea lor va fi efectuată de două institute ştiinţifice independente. Rezultatul acestei prelucrări va fi Catalogul Hipparcos, care cuprinde estimarea a 5 parametri astrometrici pentru fiecare stea din cele 100 000 care apar în catalog.

Observaţiile se efectuează prin baleiarea continuă şi sistematică a întregului cer de către telescopul dublu, care va măsura cu precizie unghiul dintre două stele îndepărtate. Prin intermediul unui set complex de oglinzi, telescopul va reflecta două cîmpuri de vedere asupra cerului separate printr-un unghi γ, în ansamblul planului focal unic care conţine o grilă periodică şi un tub proiector de imagine. Tubul este situat dincolo de planul focal şi are o arie mică de sensibilitate, permiţînd observarea unei singure stele, din mulţimea de stele vizibile simultan prin intermediul telescopului CVI, care se mută rapid de la o stea la alta în cele două cîmpuri de vedere ale telescopului pentru a face observaţii simultane. Prelucrarea datelor necesită transformarea coordonatelor corpului ceresc, pentru trecerea din planul focal al telescopului într-un sistem de referinţă inerţial obişnuit. Pentru a realiza corect această transformare, trebuie reconstituită poziţia satelitului în raport cu cele 3 axe pe baza datelor furnizate de senzorii de poziţie de la bord : cartograful stelar şi giroscoapele. Reconstituirea poziţiei este formulată ca o problemă de estimare de tip Gauss-Markov, care va fi rezolvată numai pe baza prelucrării datelor furnizate de cartograful stelar.

Reconstituirea poziţiei a fost testată prin simularea numerică a dinamicii satelitului Hipparcos şi aplicarea a 10 pulsuri de reglare pe fiecare rotaţie de spin. Poziţia a fost modelată pe baza răspunsurilor date de sistem la impulsurile de reglare aplicate, şi utilizînd serii Fourier de ordinul 20. Estimatorul Gauss-Markov a fost implementat utilizînd tehnica Householder. Pentru prelucrarea unui set de date obţinute în urma unor observaţii de durată mai lungă (mai mult de o rotaţie de spin), se vor folosi serii Fourier cu număr mai mare de termeni, acesta fiind un instrument pentru dezvoltarea ulterioară a acestei teme de cercetare ştiinţifică.

Rezultatele finale pun în evidenţă precizia foarte bună cu care se realizează reconstituirea poziţiei pe baza estimaţiei Gauss-Markov.

3. ALGORITMI ŞI SOFTWARE PENTRU APLICAŢII SPAŢIALE

Lucrările prezentate pînă acum au abordat probleme legate de reglarea poziţiei navei spaţiale. Un alt grup de lucrări, care abordează de asemenea probleme din domeniul aplicaţiilor spaţiale, este constituit de lucrările care au fost prezentate în cadrul secţiunii 12.0/B. Acestea prezintă algoritmi şi software pentru aplicaţii spaţiale.

Astfel, *lucrarea 12.0/B-1* se ocupă cu problema reglării antenelor destinate transferului de date prin intermediul sateliţilor şi se prezintă modul în care se face estimarea elementelor orbitale cu ajutorul filtrului Kalman.

Aceste antene asigură urmărirea semnalului luminos al satelitului cu ajutorul unui receptor urmăritor, lucrînd în modul de urmărire automată. Antenele pentru transmisie-recepţie de date prin intermediul sateliţiior artificiali ai Pămîntului sînt proiectate sub forma unui reflector parabolic montat astfel încît să se poată roti în jurul a două axe (azimut şi elevaţie). Datorită lăţimii reduse a fasciculului de raze, antena trebuie poziţionată cu o precizie foarte mare. De aici decurge necesitatea de a asigura un grad mare de rigiditate a antenei şi o reglare exactă a poziţionării, obţinîndu-se în acest fel o stabilitate bună în raport cu factorii perturbatori (vîntul etc.).

O problemă tehnică importantă a constituit-o sistemul de reglare pentru poziţionarea antenei. Iniţial antenele erau echipate cu un servosistem numeric pentru reglarea exactă a poziţionării, dar recent a fost adăugat un nou calculator, denumit calculator orbital.

Antena trebuie să asigure localizarea satelitului la orizont, urmărirea acestuia pe orbită şi regăsirea satelitului dacă este pierdut din vedere datorită unor perturbaţii. Pentru aceasta antena este prevăzută cu următoarele moduri de operare :

1) PRESET — antena este adusă într-o poziţie de referinţă pe baza unui algoritm optimal în timp.
2) RATE — antena primeşte referinţă de viteză pentru fiecare dintre cele două axe (azimut şi elevaţie), pînă la apariţia unei comenzi de stop.

3) SEARCH — antena face mişcări în spirală în jurul unei poziţii de pornire în scopul de a localiza satelitul.

4) AUTO-TRACK — în conformitate cu semnalul luminos sosit de la satelit, antena va primi informaţii exacte în legătură cu deviaţia dintre propria sa poziţie şi cea a satelitului. Pe baza acestor informaţii se asigură închiderea unui ciclu de reglare pentru urmărirea antenei.

5) PROGRAM-TRACK — antena urmăreşte o curbă de poziţie stabilită de către calculatorul gazdă cu ajutorul unei funcţii de timp.

Comanda antenei este realizată prin aşa-numitul servocalculator. Modurile de operare ale antenei pot fi selectate de la un panou de comandă, sau prin intermediul calculatorului gazdă. Valorile dorite pot fi introduse manual. Transferul datelor de PROGRAM-TRACK de la calculatorul gazdă la servocalculator se face cu ajutorul unei interfeţe.

Calculatorul gazdă are posibilitatea de a obţine în fiecare moment date referitoare la poziţia antenei iar în modul de operare AUTO-TRACK, poate obţine date referitoare la poziţia satelitului. Calculatorul gazdă poate de asemenea să precalculeze poziţia satelitului utilizînd modelul orbitei şi să furnizeze aceste valori servocalculatorului sub forma datelor de PROGRAM-TRACK. Combinarea acestor două sensuri de transfer permite operarea autonomă a antenei. În acest caz, calculatorul conectat la servocalculator se va numi calculator orbital.

Modelul orbital foloseşte un set de elemente orbitale care se pot modifica în timp. Dacă la un moment dat sînt cunoscute elementele orbitale, calculatorul orbital va putea calcula poziţia corespunzătoare a satelitului cu ajutorul modelului orbitei. Cînd antena a localizat satelitul, poziţiile măsurate ale azimutului şi elevaţiei vor fi utilizate pentru corectarea elementelor orbitale cu ajutorul filtrului Kalman.

Orbita satelitului este caracterizată de următoarele elemente orbitale : înclinarea, ascensiunea dreaptă, argumentul periheliului, anomalia medie în raport cu perioada, axele principale şi excentricitatea elipsei pe care se deplasează satelitul.

În modul de operare AUTO-TRACK se asigură în timp real îmbunătăţirea continuă a elementelor orbitale cu ajutorul unui filtru Kalman. Se furnizează în acest fel datele de referinţă necesare pentru poziţionarea antenei. Un astfel de calcul al datelor de referinţă devine necesar în cazul în care se cercetează orizontul în vederea localizării satelitului, cînd orbita satelitului este foarte aproape de zenitul antenei, precum şi în cazul în care trebuie asigurată redundanţa sistemului de urmărire automată. Programul a fost testat pe un minicalculator.

Lucrarea 12.0/B-2 se ocupă cu formularea filtrării prin metoda rădăcinilor de ordinul 2 pentru navigarea autonomă a satelitului.

Problema de determinare a efemeridelor şi poziţiei sateliţilor orbitali ai Pămîntului este formulată ca o problemă de filtrare cu rădăcini de ordinul 2 utilizîndu-se pentru aceasta un vector de baleiere dinspre satelit spre PCS (puncte de control la sol). Acest vector este obţinut pe baza datelor furnizate de imaginile detectate la mare distanţă. Efemeridele şi poziţia sateliţilor orbitali ai Pămîntului au fost estimate pe baza datelor furnizate de staţiile de urmărire radar de la sol, de urmăritorii de stele, de senzorii solari şi/sau tereştri etc. Precizia de estimare, este mai mare în cazul în care sînt disponibile date furnizate de imagini detectate la distanţă, PCS sînt puncte de reper cu localizare precis cunoscută pentru datele furnizate de imagini detectate la mare distanţă şi sînt utilizate pentru corectarea geometrică a acestora. Procesul de găsire a PCS din datele detectate la mare distanţă, poartă numele de armonizare a datelor. După ce s-a efectuat armonizarea datelor, se obţine determinarea unui vector de baleiere pe baza căruia se va realiza o metodă de navigare autonomă.

În lucrare este prezentată o metodă de estimare a stării utilizînd informaţia PCS obţinută pe baza datelor detectate la mare distanţă în conjuncţie cu ieşirile giroscoapelor de la bord. Se face analiza performanţelor sistemului propus pentru diferite poziţii ale PCS. Pentru astfel de probleme se poate utiliza filtrul Kalman extins. În lucrare se utilizează un algoritm al rădăcinilor pătrate, deoarece algoritmii de filtrare linearizaţi tind să devină instabili. Rezultatele obţinute pun în evidenţă faptul că filtrul poate avea performanţe foarte bune şi că algoritmul rădăcinilor pătrate este avantajos din punct de vedere numeric.

Lucrarea pune în evidenţă faptul că desfăşurarea PCS în imaginea baleiatorului multispectral are un efect considerabil asupra preciziei de estimare. Metoda prezentată poate fi utilizată în diferite probleme de navigare autonomă a sateliţilor.

4. CONCLUZII

În cadrul *secțiunii 12.0* s-au prezentat lucrări ale căror teme constituie aplicații în domeniul astronauticii.

Lucrările prezentate în cadrul primei sesiuni 12.0/A au abordat probleme legate de reglarea poziției navelor spațiale. Astfel lucrarea *12.0/A-2* propune un sistem de reglare automată a poziției unui satelit astrometric cu rotire stabilizată, pe baza schemei momentului de torsiune magnetic, în timp ce lucrarea *12.0/A-3* prezintă un sistem de reglare numerică în buclă închisă a poziției unui satelit astrometric, acționarea fiind asigurată pe bază de jeturi de gaz.

Lucrările *12.0/A-5* și *12.0/A-6* tratează probleme legate de reglarea poziției pentru cazul particular al navelor spațiale cu structură elastică.

Lucrarea 12.0/A-4 are ca temă problema reconstituirii la sol a poziției unor corpuri cerești pe baza datelor colectate în timpul unei misiuni în spațiu. Se prezintă și instrumentul optic utilizat pentru observarea corpurilor cerești, în timp ce lucrarea *12.0/A-1* are ca temă principală prezentarea unui nou observator astronomic de mare capacitate și durată mare de viață.

În cadrul sesiunii *12.0/B* s-au prezentat algoritmi și software pentru aplicații spațiale. Lucrarea *12.0/B-1* tratează problema reglării unei antene destinate transferului de date prin intermediul sateliților, prezentîndu-se și aspecte legate de estimarea elementelor orbitale cu ajutorul filtrului Kalman. Lucrarea *12.0/B-2* prezintă formularea filtrării prin metoda rădăcinilor de ordinul 2 pentru navigarea autonomă a sateliților.

Lucrările prezintă realizări practice sau experimentale de laborator care dau rezultate concludente cu privire la eficacitatea soluțiilor propuse. Se face de asemenea o testare a rezultatelor prin metode de simulare pe calculator.

Ciclul

„Service pentru calculatoare"

TESTĂRI AUTOMATE PENTRU MINICALCULATOARE

Ing. *Toma Geber*
(Coordonarea ciclului)
Ing. *Ioana Dobre*
Ing. *Camelia Busuioc*
Ing. *Dan Horhoianu*
Ing. *Mariana Niță*
Ing. *Gina Olaru*
(I.T.C.)
Ing. *Mihai Stănciulescu*
(I.I.R.U.C.)

I, SISTEM DE GENERARE DE SECVENȚE DE TEST (SGST)

CUPRINS

1. INTRODUCERE

Sistemul de Generare de Secvențe de Test (SGST) este un instrument comod și util pentru întreținerea și depanarea echipamentelor periferice ale minicalculatoarelor din familiile INDEPENDENT și CORAL. Punînd la dispoziția utilizatorului un set puternic de comenzi și un

limbaj flexibil şi uşor de învăţat, sistemul permite definirea, introducerea şi executarea diferitelor secvenţe de test pe care depanatorul şi le doreşte.

SGST poate fi inclus în sistemele de testare din familia XXDP, cu care este perfect compatibil. Faţă de programele de test autonome care testează un singur echipament la un moment dat, SGST prezintă avantajul că poate gestiona iniţierea de schimburi simultane pe mai multe periferice, simulînd condiţii de test foarte apropiate de cele din exploatarea reală. De fapt, este unicul instrument cu care se pot scrie, simplu şi rapid, secvenţe de test specifice unei situaţii reale.

Printre alte avantaje, menţionăm faptul că SGST se dovedeşte deosebit de util în depanarea echipamentelor periferice ale minicalculatoarelor lipsite de becuri şi switch-uri pentru date şi adrese pe panoul frontal. Sistemul permite atît definirea unor bucle de test scurte, destinate vizualizării pe osciloscop a unor semnale, cît şi scrierea unor programe mai complexe, care să interacţioneze, pentru acelaşi periferic sau pentru periferice diferite, punînd în evidenţă defecte mai subtile, care se manifestă sub forma conflictelor de acces.

Configuraţia hardware minimală necesară folosirii SGST constă din

— unitate centrală ;
— memorie ;
— consolă ;
— periferic suport pentru SGST (disc, bandă magnetică) ;
— echipamente periferice de testat.

2. DESCRIEREA SISTEMULUI DE GENERARE DE SECVENŢE DE TEST

2.1. Funcţiile SGST

Pentru introducerea secvenţelor de test, SGST furnizează utilizatorului un limbaj compus din instrucţiuni de uz general şi instrucţiuni de Intrare-Ieşire de nivel înalt.

Cu ajutorul acestora din urmă se pot scrie secvenţe de test specificîndu-se numai funcţia de I/E dorită, controlul la nivelul interfeţei perifericului rămînînd transparent utilizatorului. Cu ajutorul instrucţiunilor de uz general, utilizatorul poate exercita efectiv controlul asupra echipamentului, la nivel de interfaţă.

Pe lîngă acest limbaj utilizatorul dispune de un set de comenzi care îi permit folosirea funcţiilor SGST :

1. — introducerea programelor sursă ;
2. — compilarea programelor sursă ;
3. — executarea programelor obiect ;
4. — modificarea programelor sursă ;
5. — salvarea programelor sursă pe un suport extern, în vederea utilizării lor ulterioare ;
6. — încărcarea programelor sursă de pe un suport extern pentru a fi modificate, executate etc.

2.2. Structura SGST

Structura SGST este următoarea :

1. — monitorul SGST, constînd din :
   a) — executiv ;
   b) — procesor de comenzi ;
   c) — compilator ;
   d) — dispecer ;
2. — rutine de control specifice echipamentelor periferice.

2.3. Programele utilizator

Un exemplu de program utilizator este prezentat în ANEXA C.

Limbajul folosit pentru scrierea programului constă din instrucţiunile de uz general (ANEXA B) şi instrucţiunile de Intrare-Ieşire specifice echipamentului.

Numerotarea liniilor este utilă pentru modificări ulterioare și pentru salturi în program. Utilizatorul poate numerota linia în mod explicit sau SGST va incrementa automat numărul liniei cu 10, la introducerea unui blanc drept prim caracter.

Programele se individualizează printr-un număr și, opțional, prin nume.

Fiecare program își exercită controlul asupra unui singur periferic (la un moment dat) și este tratat ca o entitate separată de către SGST.

Programului îi pot fi asignate mai multe echipamente de același tip și, printr-o singură comandă RUN, testul va fi executat pentru toate echipamentele, în mod secvențial.

2.4. Gestionarea memoriei

SGST asigură posibilitatea executării simultane a 16 programe utilizator prin încărcarea acestora în 16 partiții distincte, a căror dimensiune este stabilită dinamic.

Structura de principiu a unei partiții :

Tabela Informații Program (TIP)
Rutina de control a echipamentului (RCE)
Zona de Date
Program utilizator format sursă
Program utilizator format executabil

TIP (Tabela Informații Program) conține :

1. — vectorul de stare al programului (PFLGWD) ;
2. — switch-urile de operare pentru program (POPSW) ;
3. — numărul și numele programului ;
4. — tipul echipamentului ;
5. — numărul echipamentului curent testat ;
6. — numerele echipamentelor asignate spre a fi testate ;
7. — celulele de lucru TEM0-TEM15 ;
8. — alte celule de lucru.

Rutina de control a echipamentului îndeplinește funcția unui driver, permițînd execuția instrucțiunilor de Intrare-Ieșire de nivel înalt, specifice perifericului. În plus, asigură afișarea erorilor (prin mesaje explicite), a registrelor de stare și a unor informații statistice.

Deoarece multe instrucțiuni de Intrare-Ieșire nu depind de o anumită zonă de memorie, SGST asigură două buffer-e de date cu nume simbolice predefinite (RDIO -citire, WRIO -scriere), localizate în zona de date din partiție. Dimensiunile buffer-elor au valoarea implicită de 256 bytes, dar pot fi schimbate, în momentul introducerii programului, cu orice valoare între 0 și 32 776.

Utilizatorul mai dispune de o altă zonă de I/E, cunoscută sub numele de FREE, care se extinde de la sfîrșitul SGST pînă la sfîrșitul memoriei. Ea se folosește pentru operații de I/E de mare anvergură și ca zonă comună pentru transfer de date între programe diferite (vezi exemplul din ANEXA C).

2.5. Alte facilităţi de scriere a programelor

2.5.1. *SWITCH-URI DE OPERARE*

Switch-urile de operare, grupate într-un vector numit OPSW, se referă la funcţii comune tuturor programelor scrise cu SGST :

— controlul execuţiei programului în diverse situaţii (în caz de eroare, cînd programului îi sînt asignate mai multe unităţi de acelaşi tip) ;

— controlul tipăririi mesajelor (mesaje de eroare, informaţii statistice) ;

— execuţia unor funcţii speciale (lucrul în modul maintenance etc.).

Semnificaţia biţilor OPSW este dată în ANEXA D.

2.5.2. *OPERANZI PREDEFINIŢI*

SGST pune la dispoziţia utilizatorului posibilităţi de generare a unor cîmpuri de date, de lungime variabilă, prin folosirea diferitelor combinaţii de operanzi predefiniţi, identificaţi prin nume simbolice (ANEXA E) sau prin folosirea operandului RANDOM.

2.6. Raportarea erorilor

Mesajele de eroare pot fi grupate în trei categorii :

1. Mesajele de eroare ale procesorului de comenzi, care se referă la :
   a) erori de sintaxă ale comenzilor ;
   b) erori detectate în timpul execuţiei comenzilor ;
   c) erori detectate în timpul execuţiei programelor.
2. Mesajele de eroare ale compilatorului, care se referă la sintaxa incorectă a liniilor din programul sursă al utilizatorului.
3. Mesajele de eroare ale rutinelor de control, care se referă la :
   1. erori de date ;
   2. erori propriu-zise de Intrare-Ieşire.

3. COMPONENTELE SGST

3.1. Monitorul

3.1.1. *EXECUTIVUL*

Executivul realizează următoarele :

— iniţializarea memoriei şi a tabelelor SGST ;

— încarcă fişierul TVDAnm.SGT, care conţine lista perifericelor admise de SGST ;

— tipăreşte informaţii ca :

  — punctele de restart ;

  — limitele zonei FREE.

— transferă controlul procesorului de comenzi.

3.1.2. *PROCESORUL DE COMENZI*

Procesorul de comenzi realizează următoarele :

— analizează linia de comandă ;

— detectează şi semnalează eventualele erori de sintaxă ;

— execută comanda (comenzile RUN, STOP, CONT presupun cedarea controlului între procesorul de comenzi, progamul utilizator, şi dispecer) ;

— semnalează eventualele erori survenite în timpul execuţiei comenzii.

Comenzile pot fi împărţite în patru categorii:

1. comenzi asociate programului utilizator — de introducere, lansare în execuţie, oprire, relansare, ştergere etc.;
2. comenzi de generare de cîmpuri de date;
3. comenzi de lucru cu programele sursă — editare, salvare, încărcare, afişare;
4. comenzi utilitare — conversii, calcul de offset, afişare şi modificare conţinut celule de memorie.

Lista comenzilor este dată în ANEXA A.

3.1.3. *COMPILATORUL*

Compilatorul asigură traducerea liniilor sursă în cod obiect, direct executabil. Detectează şi afişează eventualele erori de sintaxă din programul sursă.

Instrucţiunile recunoscute de compilator sînt cele de uz general (ANEXA B) şi cele de I/E asigurate de rutinele de control (vezi secţiunea 3.2).

3.1.4. *DISPECERUL*

Dispecerul asigură execuţia simultană a maximum 16 programe utilizator, permiţînd evidenţierea eventualelor conflicte de acces la resursele sistemului sau depanarea simultană a mai multor periferice.

Dispecerul scanează partiţiile, folosind o tehnică tip polling, pentru a afla dacă un program este prezent, a fost specificat pentru a fi executat şi îndeplineşte condiţiile necesare lansării în execuţie. Dacă programul nu a fost oprit de utilizator, nu a detectat o eroare şi nu aşteaptă terminarea unei operaţii de I/E, atunci dispecerul transferă controlul programului obiect la adresa lui curentă. Programul păstrează controlul pînă cînd apare unul din evenimentele următoare, caz în care el va reda controlul dispecerului:

1. a fost iniţiată o operaţie de I/E şi se aşteaptă sfîrşitul ei;
2. se încearcă lansarea unei operaţii de I/E, dar perifericul e ocupat;
3. s-a detectat o eroare pe durata unei operaţii de I/E;
4. programul utilizator conţine o instrucţiune LET GO (cedează controlul);
5. programul utilizator s-a terminat.

După ce a reprimit controlul, dispecerul realizează operaţiile necesare pentru a suspenda programul şi testează următoarea partiţie. Cînd a terminat scanarea ultimei partiţii va reveni la prima.

Dacă toate programele aşteaptă terminarea unor operaţii de I/E, dispecerul va continua, în buclă, testarea partiţiilor căutînd un program care a devenit gata pentru reluarea execuţiei.

Dacă o operaţie de I/E se termină, programul care a lansat-o va fi reluat la următoarea scanare a partiţiei asociate. Dacă se încearcă lansarea unei operaţii de I/E pe un periferic care este deja ocupat sau programul execută instrucţiunea LET GO (cedează controlul), scanarea pentru lansarea în execuţie a altor programe se reia cu partiţia următoare, aceasta permiţînd ca fiecare program să primească controlul măcar o dată.

Scanarea partiţiilor continuă pînă cînd fie toate programele s-au terminat, fie ele au fost oprite de utilizator sau de apariţia unei erori.

3.2. Rutine de control

3.2.1. *CARACTERISTICI GENERALE*

Execuţia instrucţiunilor de Intrare-Ieşire de nivel înalt, specifice perifericului, este asigurată de prezenţa rutinei de control respective. În lipsa acesteia, întreaga pregătire a execuţiei unei operaţii de Intrare-Ieşire este asigurată de utilizator în programul său, folosind setul de instrucţiuni de uz general.

Rutina de control este o colecţie de tabele şi subrutine care gestionează echipamentul la nivel de interfaţă.

INSTRUCŢIUNI

Pe lîngă instrucţiunile specifice fiecărui periferic, care vor fi descrise în paragrafele următoare, toate rutinele de control asigură execuţia instrucţiunilor de mai jos :

NOWAIT

După iniţierea unei operaţii de Intrare-Ieşire nu se aşteaptă terminarea ei, ci se continuă execuţia programului utilizator cu instrucţiunea următoare. Acest lucru permite lansarea altei operaţii de I/E, vizualizarea registrelor şi/sau alte prelucrări de date în acest timp. Pentru periferice de teletransmisie, acest mod de lucru permite execuţia simultană de citiri şi scrieri pentru verificarea tuturor liniilor.

WAIT

După lansarea unei operaţii de Intrare-Ieşire se aşteaptă terminarea acesteia înainte de executarea instrucţiunii următoare din programul sursă.

COUNTS

Se afişează informaţiile statistice.

STATUS

Se afişează conţinutul curent al registrelor perifericului şi conţinutul lor la ultima întrerupere.

INFORMAŢII STATISTICE

Informaţiile statistice sînt stocate în nişte celule de memorie şi contorizează :
— numărul de octeţi transferaţi ;
— numărul de comenzi de acelaşi fel executate de program ;
— numărul de intrări în rutina de întrerupere ;
— numărul de erori.

CUVINTE DE INTERFAŢĂ CU PROGRAMUL UTILIZATOR

Fiecare rutină de control are la dispoziţie un număr de opt cuvinte, dintre care şase pot fi folosite în funcţie de necesităţile proprii rutinei de control (CYL, HEAD, SECT pentru disc ; EOT, EOF pentru banda magnetică ; număr de reluări ale unei operaţii de I/E etc.). Celelalte două cuvinte au asignate nume simbolice şi sînt comune tuturor rutinelor de control :

SIZE

Numărul de octeţi transferaţi efectiv de o operaţie de I/E.

ERR

Flag de eroare, resetat la iniţierea fiecărei operaţii de I/E.

RAPORTAREA ERORILOR

În momentul detectării unei erori, minimul de informaţii afişate va fi :
— identificatorul erorii ;
— numărul liniei din programul sursă generatoare de eroare.

În funcţie de tipul erorii pot fi afişate şi alte informaţii ca :
— numărul unităţii curente ;
— conţinutul registrelor perifericului ;
— mnemonicele biţilor de eroare.

După raportarea erorii nu se mai reia execuţia programului decît dacă bitul 15 din OPSW este setat sau dacă se dă comanda CONT.

3.2.1. *Rutina de control RK05*

RUTINA DE CONTROL TRKAnn.SGT gestionează operaţiile de intrare/ieşire pentru discul RKO5 conectat prin cuptor de tip RK11 (cartridge).

Informaţiile specifice echipamentului sînt :

— mnemonica : RK11 sau RK05 (se acceptă ambele) ;

— număr de unităţi : 0—7 ;

— adresa de bază : 177400 ;

— adresa vectorului de întrerupere : 220 ;

— prioritate : 5.

Numele simbolice ale registrelor cuplorului şi deplasamentele lor faţă de adresa de bază sînt :

NUME	DEPLASAMENT	FUNCŢIE
1. RKDS	0	Drive Status Register
2. RKER	+2	Error Register
3. RKCS	+4	Control and Status Register
4. RKWC	+6	Word Count Register
5. RKBA	+10	Bus Address Register
6. RKDA	+12	Disk Address Register
7. RKDB	+16	Data Buffer Register

Instrucţiunile de nivel înalt specifice echipamentului RK05 se împart în :

a) instrucţiuni prin care se lansează o operaţie de I/E ;

b) instrucţiuni prin care nu se lansează o operaţie de I/E.

a) Instrucţiunile de nivel înalt prin care se lansează o operaţie de I/E se pot efectua cu valori implicite sau cu valori definite de utilizator (pentru lungimea blocului de date transferat, pentru adresa discului şi adresele de memorie între care are loc transferul). Dacă se folosesc valorile implicite, atunci citirea se face în bufferul de citire (RDIO), iar scrierea în bufferul de scriere (WRIO) din partiţie, pe o lungime de 256 octeţi. Adresa discului este indicată în cuvintele de interfaţă CYL, HEAD, SECT.

Instrucţiunile sînt următoarele :

READ (D256 INTO RDIO)
READ n (INTO RDIO)
READ n INTO a

Se transferă n octeţi (implicit 256) de pe disc în memorie, la adresa a (implicit în bufferul RDIO).

WRITE (D256 FROM WRIO)
WRITE n (FROM WRIO)
WRITE n FROM a

Se transferă n octeţi (implicit 256) din memorie de la adresa a (implicit buffer WRIO) pe disc.

SEEK

Se lansează funcţia SEEK la adresa de disc indicată în cuvintele de interfaţă CYL, HEAD, SECT. Se aşteaptă întreruperea „Search Complete".

WRLOCK

Unitatea testată va fi protejată la scriere.

RDFMT n INTO a

Se lansează funcţia READ cu bitul de format FMT=1. Se transferă deci doar cîte 2 octeţi pe sector (header). Sintaxa nu permite valori implicite pentru n şi a.

WRFMT n FROM a
: Se lansează operaţia WRITE cu bitul FMT=1. Se scriu cîte 2 octeţi pentru fiecare din cele n sectoare, de la adresa a din memorie. Nu se acceptă valori implicite pentru n şi a.

RDCK
: Se lansează funcţia READ CHECK la sectorul a cărui adresă este în cuvintele de interfaţă CYL, HEAD, SECT.

WRCK n AT a
: Se lansează operaţia WRITE CHECK cu n octeţi la adresa de memorie a. u se admit valori implicite.

CRESET
: Se dă comanda CONTROL RESET cuplorului.

DRESET
: Se dă comanda DRIVE RESET unităţii curent testate şi se aşteaptă întreruperea "Search Complete".

b) Instrucţiunile de nivel înalt care nu lansează o operaţie de intrare/ieşire sînt :

STEPUP n
: Se creşte adresa disc cu n sectoare, furnizîndu-se totodată o adresă de disc validă (cilindru, eap, sector)

STEPDN n
: Se scade adresa disc cu n sectoare, generîndu-se totodată o adresă de disc validă (cilindru, cap, sector).

3.2.2. *RUTINA DE CONTROL TM02/TU16*

RUTINA DE CONTROL TMMAnn.SGT gestionează operaţiile de intrare/ieşire pentru banda TU16 conectată prin cuplor de tip TM02.

Informaţiile specifice echipamentului sînt :

— mnemonica : TM02 sau TU16 (se acceptă ambele) ;
— număr de unităţi : 0—77 (8) ;
— adresa de bază : 172440 ;
— adresa vectorului de întrerupere : 224 ;
— prioritate : 5.

Numele simbolice ale registrelor cuplorului şi deplasamentele lor faţă de adresa de bază sînt :

NUME	DEPLASAMENT	FUNCŢIE
1. MTC1	+0	Control Status 1 Register
2. MTWC	+2	Word Count Register
3. MTBA	+4	Bus Address Register
4. MTFC	+6	Frame Count Register
5. MTC2	+10	Control Status 2 Register
6. MTDS	+12	Drive Status Register
7. MTER	+14	Error Register
8. MTAS	+16	Attention Summary Register
9. MTDB	+22	Data Buffer Register
10. MTMR	+24	Maintenance Register
11. MTDT	+26	Drive Type Register
12. MTSN	+30	Serial Number Register
13. MTTC	+32	Tape Control Register

Instrucţiuni de nivel înalt specifice :

READ (D256 INTO RDIO)
READ n (INTO RDIO)
READ n INTO a
: — transferă n octeţi de pe bandă în memorie la adresa a ; banda merge înainte.

WRITE (D256 FROM WRIO)
WRITE n (FROM WRIO)
WRITE n FROM a
— se transferă n octeţi din memorie, de la adresa a, pe bandă.

READRV n INTO a
— se transferă n octeţi de pe bandă în memorie, la adresa a ; banda merge înapoi.

WRITM — scrie TAPE MARK.

WCKFWD n FROM a
— compară n octeţi de la adresa a cu cei din blocul curent de pe bandă, aceasta deplasîndu-se înainte.

WCKREV n FROM a
— similară cu WCKFWD, banda se mişcă înapoi.

SPWFD n — salt înainte peste n blocuri.
SPREV n — salt înapoi peste n blocuri.
REWIND — rebobinare pînă la BOT.
RWDOFL — rebobinare cu trecere OFF-LINE.
DCLEAR
— iniţializare cuplor-formatter-drive.

ERASE
— şterge aproximativ 7 cm de bandă.

ODD — seteaza mod paritate impară
EVEN — seteaza mod paritate pară.
BPI v
— modifică densitatea de scriere pe bandă, conform valorii v :
200— 200 bpi
556— 556 bpi
800— 800 bpi
1 600—1 600 bpi

FMT v
— modificarea formatului de scriere pe bandă, conform valorii v :
N — NORMAL
C — CORE DUMP
M — 15 MODE

MWRITE V
— verificarea lanţului de scriere în mod maintenance, folosind cuvîntul de date de la adresa v.

MREAD v
— verificarea lanţului de citire în mod maintenance, folosind cuvîntul de date de la adresa v.

3.2.3 *RUTINA DE CONTROL RX11/RX01*

Rutina de control RTXAnn.SGT gestionează operaţiile de intrare/ieşire pentru discul flexibil simplă densitate RX01 conectat prin cuplor de tip RX11.

Informaţiile specifice echipamentului sînt :
— mnemonica : RX11 sau RX01 (se acceptă ambele) ;
— număr de unităţi : 2 ;
— adresa de bază : 177170 ;
— adresa vectorului de întrerupere : 264 ;
— prioritate : 5.

Numele simbolice ale registrelor cuplorului şi deplasamentele lor faţă de adresa de bază sînt :

NUME	DEPLASAMENT	FUNCŢIE
1. RXCS	0	Control Status Register
2. RXDB	+2	Data Register
3. RXTA	+2	Track Address Register
4. RXSA	+2	Sector Address Register
5. RXES	+2	Error & Status Register

Zona de interfaţă cu programul utilizator constă din următoarele locaţii:

TRAK
Un cuvînt conţinînd numărul pistei în biţii 0—6, număr ce va fi folosit în următoarele operaţii deI/E.

SECT
Un cuvînt conţinînd numărul sectorului în biţii 0—4, număr ce va fi folosit în următoarele operaţii de I/E.

RTRY
Numărul de reluări ale operaţiei de I/E înainte de a decide eroare nerecuperabilă.

SIZE, ERRI
Implementare standard.

Instrucţiunile de nivel înalt specifice echipamentului RX01 sînt următoarele:

READ (D256 INTO RDIO)
READ n (INTO RDIO)
READ n INTO a
Se citesc n octeţi (implicit 256) la adresa de memorie a (implicit RDIO).

WRITE (D256 FROM WRIO)
WRITE n (FROM WRIO)
WRITE n FROM a
Se scriu n octeţi (implicit 256) de la adresa de memorie a (implicit WRIO).

CRESET
Controller Clear (RESET).

STEPUP n
Incrementează valoarea SECT cu n, generînd o adresă validă de sector şi pistă.

STEPDN n
Decrementează valoarea SECT cu n, generînd o adresă validă de sector şi pistă.

WRDDS
Marchează sectorul şters.

FILBF n FROM a
Se scriu n octeţi de la adresa a în SILO (max. 128 octeţi).

EMPTYB n INTO a
Se citesc n octeţi din SILO în memorie la adresa a.

WRSEC
Se scrie un sector pe discul flexibil. Adresa discului este specificată de locaţiile TRAK şi SECT.

RDSEC
Se citeşte un sector de pe discul flexibil. Adresa discului este specificată de locaţiile TRAK şi SECT.

RDSTAT
Se afişează registrul de stare al echipamentului.

RDERR
Se afişează codul de eroare.

Observaţie : Dacă bitul 7 (200) din OPSW este setat, codul erorii va fi afişat (cu mesajul explicativ) la orice eroare nerecuperabilă.

3.2.4 *RUTINA DE CONTROL DH11*

Rutina de control TDHAnn.SGT gestionează operaţiile de intrare/ieşire pentru multiplexorul asincron programabil pe 16 linii DH11.

Informaţiile specifice echipamentului sînt :

— mnemonica : DH11 ;
— adresa de bază : 160020 ;
— adresa vectorului de întrerupere : 300 ;
— prioritate : 5 ; 5.

Numele simbolice ale registrelor cuplorului şi deplasamentele lor faţă de adresa de bază sînt :

NUME	DEPLASAMENT	FUNCŢIE
1. SCR	0	System Control Register
2. NRC	+2	Next Received Character
3. LPR	+4	Line Parameter Register
4. CAR	+6	Current Address Register
5. BYCR	+10	Byte Count Register
6. BAR	+12	Buffer Active Register
7. BCR	+14	Break Control Register
8. SSR	+16	Silo Status Register

Instrucţiunile de nivel înalt specifice echipamentului DH11 sînt următoarele :

READ (D256 INTO RDIO)
READ n (INTO RDIO)
READ n INTO a FROM u
READ n INTO a

Recepţie n caractere (implicit 256).

WRITE (D256 FROM WRIO)
WRITE n (FROM WRIO)
WRITE n FROM a FROM u
WRITE n FROM a

Emisie n caractere (implicit 256).

BREAK t
BREAK t ON l 1 [, ... l i . . .]

Se generează o întrerupere de durata t pe liniile l 1, ... l i.

CRESET

Iniţializare cuplor.

ALARM v

Se încarcă nivelul de alarmă SILO (v poate fi 0, 1, 2, 4, 10, 20, 40).

SETUP l [l 1, ...]

Selectează liniile pentru care se vor stabili parametri în LPR cu instrucţiunile de mai jos.

RBAUD v

Selectează viteza de recepţie v :

v	
v=0	0
v=1	50
v=2	75
v=3	110
v=4	134,5
v=5	150
v=6	200
v=7	300
v=10	600
v=11	1200
v=12	1800
v=13	2400
v=14	4800
v=15	9600
v=16	EXT A
v=17	EXT B

TBAUD v
Selectează viteza de transmisie v.

BAUD v
Selectează viteza de transmisie/recepție v.

EVEN
Selectează paritate pară.

ODD
Selectează paritate impară.

NOPAR
Dezactivează paritatea.

ONESTP
Setează numărul de biți de stop = 1.

TWOSTP
Setează numărul de biți de stop = 2.

BITS n
Selectează numărul de biți pe caracter (n poate fi 5, 6, 7, 10 sau D8).

ECHO
Activează auto-ecoul.

NOECHO
Dezactivează auto-ecoul.

PRESET
Selectează ca parametri: 8 biți pe caracter, fără paritate, 2 biți de stop, 110 bauds, fără ecou, full duplex.

FDUPLX
Selectează mod full duplex.

HDUPLX
Selectează mod half duplex.

Observație : Dacă bitul 7 (SPOPER) din OPSW este setat, se va pune pe 1 bitul de maintenance (bitul 9) din SCR.

3.2.5. *RUTINA DE CONTROL DL11*

Rutina de control TDLAnn.SGT gestionează operațiile de intrare/ieșire pentru interfața de linie simplă serială asincronă DL11.

Informațiile specifice echipamentului sînt :

— mnemonica : DL11 ;
— adresa de bază : 175610 ;
— adresa vectorului de întrerupere : 300 ;
— prioritate : 4 ; 4.

Numele simbolice ale registrelor cuplorului și deplasamentele lor față de adresa de bază sînt :

NUME	DEPLASAMENT	FUNCȚIE
1. RCSR	0	Receiver Status Register
2. RBUF	+2	Receiver Data Buffer
3. XCSR	+4	Transmitter Status Register
4. XBUF	+6	Transmitter Data Buffer

Instrucțiunile de nivel înalt specifice echipamentului DL11 sînt următoarele :

READ (D256 INTO RDIO)
READ n (INTO RDIO)
READ n INTO a
Recepție n caractere (implicit 256).

WRITE (D256 FROM WRIO)
WRITE n (FROM WRIO)
WRITE n FROM a
Transmisie n caractere (implicit 256).

BREAK t
Generează întrerupere de durată t.

CRESET
Iniţializare cuplor.

CALL
Iniţiază un apel către DL11.

LISTEN
Aşteaptă un apel de la DL11.

ANSWER
Răspunde unui apel de la DL11.

HANGUP
Termină un apel

SEND
Setează linia pentru transmisie.

RECV
Setează linia pentru recepţie.

RDRON
Setează bitul 0 din RCSR care deschide releul de pornire al benzii de hîrtie (unde e posibil).

RDROFF
Şterge bitul 0 din RCSR.

Observaţie : Dacă bitul 7 (SPOPER) din OPSW este setat, atunci se va seta bitul 2 din XCSR (maintenance).

3.2.6. *RUTINA DE CONTROL LP11*

RUTINA DE CONTROL TLPAnn.SGT gestionează operaţiile de intrare/ieşire pentru imprimanta de tip LP11.

Informaţiile specifice echipamentului sînt :

— mnemonica : LP11 ;
— adresa de bază : 177514 ;
— adresa vectorului de întrerupere : 200 ;
— prioritate : 4.

Numele simbolice ale registrelor cuplorului şi deplasamentelor faţă de adresa de bază sînt :

NUME	DEPLASAMENT	FUNCŢIE
1. LPS	0	Control Status Register
2. LPB	+2	Data Buffer Register

Instrucţiunile de nivel înalt specifice echipamentului LP11 sînt următoarele :

WRITE (D256 FROM WRIO)
WRITE n (FROM WRIO)
WRITE n FROM a
Se transferă n octeţi (implicit 256) din memorie de la adresa a (implicit buffer WRIO) la imprimantă. După date se inserează automat CR/LF, dacă bitul SPOPER din OPSW=O.

SPACE n
Se avansează n linii (prin scrierea în bufferul de date a două combinaţii CR/LF).

TOF
Top of Form. Se trimite la imprimantă caracterul FORM FEED (cod 014).

EOT
End of Transmission. Se trimite caracterul EOT (cod 004) către imprimantă.

Observaţie : Dacă bitul 7 (SPOPER) din OPSW=1, se inhibă inserarea automată a caracterelor CR/LF după datele transmise spre imprimantă.

4. PROCEDURA DE LANSARE ÎN EXECUŢIE ŞI OPERARE

Sistemul de Generare de Secvenţe de Test este încărcat de monitorul XXDP sau de utilitarul UPD2, la fel ca orice diagnostic. Este încărcat întîi Executivul, care realizează o iniţializare a locaţiilor 0-276 (trap catcher), determină dimensiunea memoriei prezente fizic, şterge zona de memorie de deasupra lui şi determină prezenţa unităţii de management a memoriei, prin adresarea registrului MMR0. Dacă unitatea de management există, Executivul va interoga operatorul asupra folosirii variantei cu sau fără management a SGST. În conformitate cu răspunsul primit de la acesta, Executivul va încărca monitorul TMGAnm.SGT (fără management) sau TMMAnm.SGT (cu management).

Monitorul va iniţializa o serie de locaţii şi contoare, îşi va scrie numele, versiunea şi adresa de restart şi va încărca apoi Tabela de Echipamente Permise (TVDAnm.SGT). În continuare, monitorul va afişa harta ocupării memoriei (Memory Map), va evalua zona de memorie disponibilă (FREE) şi va tipări mesajul "ENTER CMND" şi prompterul * (asterisc), aşteptînd introducerea uneia din comenzile de editare program sursă, modificare, execuţie, salvare sau restaurare de programe (vezi Anexa A).

Execuţia unui program poate fi întreruptă, revenind în monitor, prin apăsarea tastelor CTRL-C. În mod „întrerupere", monitorul acceptă doar un subset limitat de comenzi, care nu modifică programele existente, ci doar raportează erorile, schimbă switch-urile de execuţie a unuia sau a mai multor programe sau opreşte execuţia unuia sau mai multor programe. După ce s-a dat o nouă comandă RUN, execuţia se reia, prin scanarea succesivă a partiţiilor. Se continuă astfel pînă cînd s-au terminat toate programele şi s-au executat toate comenzile.

În Anexa C am prezentat cîteva exemple de introducere şi de execuţie de secvenţe de test (pentru discurile magnetice RK05, banda magnetică TM02 şi imprimanta LP11).

ANEXA A

Setul de comenzi SGST

a) Comenzi asociate programului utilizator.

ENTER p [AS nume]

— trecere în mod introducere linii sursă pentru programul p cu "nume" opţional.

RUN [p1 [AT ln] [, . . . [, pn [AT ln]]]]

— lansează în execuţie programele utilizator pn, de la prima linie sursă sau de la cea specificată prin ln.

STOP p1 [,. . ., pn]

— opreşte unul sau mai multe programe utilizator.

CONT p1 [, . . ., pn]

— reia execuţia unui sau mai multor programe utilizator oprite de o comandă utilizator sau de apariţia unei erori. Execuţia efectivă nu este reluată decît la introducerea unei comenzi RUN sau a unei linii nule.

KILL p1 [, . . ., pn]

— forţează terminarea unui program. Programul poate fi restartat cu comanda RUN.

DELETE p1 [, . . ., pn]

— şterge programele specificate şi eliberează memoria alocată lor.

ASSIGN mdl [, u1, . . ., un] TO p

— permite schimbarea numelor de unităţi asignate unui program existent şi/sau schimbarea constantelor perifericului.

ASSIGN {mdl, u TO {SAVE
{NONE {FETCH
{LIST

— asignare, deasignare a perifericului curent drept dispozitiv LIST, SAVE, FETCH.

REPORT p [, p STATUS ,p]
REPORT p [,p COUNTS ,p]

— forțează tipărirea tuturor registrelor echipamentului, informațiile de eroare și informațiile statistice pentru programul (programele) specificate. Are efect doar asupra programelor scrise pentru echipamente care au rutine de control.

OPSW [p] oooo

— folosit pentru a modifica switch-urile de operare. Dacă numărul programului nu este specificat, noua valoare a OPSW va fi valabilă pentru toate programele rezidente în memorie, altfel doar OPSW-ul programului p va fi modificat.

b) Comenzi de generare de cîmpuri de date.

FILL BFnn WITH {oooo, oooo, . . ., etc
{ * aaa . . . a
{ #aaa . . . a

— permite încărcarea celor 16 buffer-e comune de date fie cu numere octale, fie cu caractere ASCII. Valoarea nn poate fi între 00 și 15, specificînd care buffer va fi încărcat. Codurille *, # pentru formarea caracterelor ASCII permit introducerea sau nu a caracterelor de control ⟨CR⟩, ⟨LF⟩.

FILL BUF WITH {PATn
{oooo, oooo, . . . , etc.

— întreg buffer-ul de 256 bytes este încărcat fie cu o configurație de date predefinită, fie cu cuvinte în octal.

FILL COM[n] WITH {oooo
{ * aaa . . . a
{ #aaa . . . a
{PATn

— aceasta este o altă formă a comenzii FILL care permite accesul la cele 10 cuvinte comune COMO—COM9 ca și la zonele BUF00-BUF15.

c) Comenzi de lucru cu programele sursă.

MODIFY p

— următoarele linii sursă vor fi considerate corecții pentru programul p, existent deja.

DISPLAY p [, p CODE ,p]

— afișează liniile sursă ale programului specificat cu sau fără codul obiect generat.

/SAVE p [AS name]

— liniile sursă ale programului p vor fi scrise pe dispozitivul SAVE curent, cu un identificator de 1—6 caractere furnizate de "name". Dacă acesta nu este specificat, programul va fi salvat sub numele asignat acestuia în memorie. Programul nu va fi șters din memorie.

/FETCH name AS p

— citește un program salvat anterior.

/DELETE name [, name, . . ., name]

— șterge programele utilizator specificate, salvate anterior pe dispozitivul SAVE curent.

/LIST␣{ALL
␣{FAST
␣{ALL, FAST

— listeaza fișierele de pe dispozitivul FETCH.

ALL — se listează toate fișierele.

FAST — se listează doar numele fișierelor.

/ZERO

— inițializează dispozitivul SAVE curent. Cînd se încearcă inițializarea dispozitivului suport pentru SGST este afișat un mesaj de avertizare care așteaptă un răspuns afirmativ pentru a efectua inițializarea.

/BOOT

— execută funcția de bootstrap-are de pe dispozitivul FETCH curent.

d) Comenzi utilitare.

FM

— formatează memoria eliminînd orice spaţiu nefolosit care a apărut între SGST şi programele utilizator.

RDM aaaa [, cccc]

— afişează conţinutul celulei de memorie de la adresa octală aaaa. Dacă şi a doua adresă este specificată se afişează întreaga zonă de memorie de la aaaa la cccc, inclusiv.

WRM aaaa, oooo [, oooo, . . ., etc.]

— scrie un cuvînt sau un şir de cuvinte octale începînd de la adresa octală de memorie aaaa.

BOC aaaa, cccc

— calculează valoarea deplasamentului dintre două adrese pare de memorie.

ADD oooo, oooo [, oooo, . . ., etc.]

— adună două sau mai multe numere octale şi afişează rezultatul.

SUB oooo, oooo

— scade primul număr octal din al doilea şi afişează rezultatul.

CBD oooo

— converteşte un număr octal într-unul zecimal şi afişează rezultatul.

CDB dddd

— converteşte un număr zecimal într-unul octal şi afişează rezultatul.

ANEXA B

Instrucţiuni de uz general pentru scrierea programelor sursă

LOAD a WITH v

— încarcă celula de memorie de la adresa a cu valoarea v.

INCR a [BY i]

— incrementează valoarea de la adresa a cu i. Dacă i nu e specificat incrementarea se face cu 1.

DECR a [BY i]

— decrementează valoarea de la adresa a cu i. Dacă i nu e specificat decrementarea se face cu 1.

MOVE

— această instrucţiune, fără parametri, implică un „move" indirect. Numărul de bytes de transferat e conţinut de locaţia NBR, iar adresa de unde încep în locaţia SRC. Adresa destinaţiei se găseşte în locaţia DST.

MOVE [n AT] a1 TO a2

— se mută n bytes de la adresa a1 la adresa a2. Dacă n nu e specificat se mută un cuvînt.

ADD [n AT] a1 TO a2

— adună n bytes de la adresa a1 şi cu n bytes de la adresa a2 şi pune rezultatul la adresa a2. Dacă n lipseşte se adună doar cîte un cuvînt.

SUB [n AT] a1 TO a2

— se efectuează scăderea pe n bytes, descăzutul fiind la adresa a2. iar scăzătorul la adresa a1. Rezultatul este pus la adresa a2. Dacă n lipseşte se efectuează scăderea pe cuvînt.

NEGATE a

— se efectuează complementul faţă de 2 al conţinutului celulei de memorie de la adresa a.

SET a BIT n [THRU n [AND n]]
          [—   &

— se setează bitul n în locaţia a. Pot fi specificate combinaţii de biţi (AND, &) sau grupuri de biţi (THRU, —).

CLEAR a BIT n [THRU n [AND n]]
          [—   &

— se şterge bitul n de la locaţia a. Ca şi la SET se pot specifica combinaţii şi grupuri de biţi.

IF a BIT n [AND n . . . AND n [THRU n]] {SET GO TO ln
{CLEAR

— dacă bitul sau combinaţia de biţi specificată îndeplineşte condiţia, atunci se execută salt la linia sursă ln.

IF [n AT] a1=a2 GO TO ln
 \>
 <
 ⇒
 ⇐
 ⟨ ⟩

— dacă relaţia specificată între conţinutul celulelor a1 şi a2 este adevărată se execută salt la linia sursă ln ; n specifică numărul de bytes pe care se face comparaţia.

GO TO ln

— salt necondiţionat în program, la linia sursă ln.

LINK ln

— salt la subrutina care începe de la adresa ln.

RETURN

— reîntoarcere din subrutină.

PRINT * text
          #

— se tipăreşte textul specificat la dispozitivul de tipărire asignat programului. După text urmează sau nu ⟨CR⟩, ⟨LF⟩, după cum se folosesc * sau #.

PRINT [n AT] a IN {BINARY
{OCTAL
{DECIMAL
{ASCII

— se convertesc n bytes conform formatului cerut şi se tipăresc la dispozitivul de tipărire asignat programului. Dacă n lipseşte se tipăreşte un singur cuvînt.

FILL [n AT] a WITH {RANDOM
{ASCII
{v
{PATn

— se umplu n bytes, începînd de la adressa a, cu date generate aleator, cu caractere ASCII, cu combinaţia de biţi specificată sau cu pattern-ul n. Cînd n lipseşte, valoarea sa este implicit 256.

ROTATE [n AT] a

— se rotesc la dreapta cu o poziţie, n bytes de la locaţia a ; a este implicit adresa buffer-ului WRIO pentru acest program.

VECTOR a TO ln

— se memorează adresa liniei sursa ln la adresa a.

VERIFY [n AT] a1 WITH a?

— se face o comparaţie pe n bytes a două zone de memorie, prima începînd de la adresa a1, a doua de la adresa a2. Se tipăresc, în octal, perechile care nu concordă.

PAUSE

este compilată ca un WAIT.

DELAY v

— se aşteaptă v microsecunde. Variabila v trebuie să aibă o valoare zecimală.

ENTRY

— salvează în stivă registrele R0—R5 şi e folosită pentru începerea rutinei de întrerupere scrisă de utilizator.

EXIT

— reface registrele R0—R5 şi permite ieşirea din rutina de întreruperi scrisă de utilizator.

LET GO

— prin această instrucţiune programul utilizator redă controlul dispecerului.

RESET — instrucţiunea RESET.

WORD v

— se rezervă în programul utilizator un cuvînt a cărui valoare este v.

END

— marchează sfîrşitul programului. Trebuie să fie instrucţiunea cu cel mai mare număr de linie.

ANEXA C

Exemple de secvenţe de test

```
*ENTER 1 AS DISK1
?ASGN DEV : RK11, 0
?RDIO = 256/0
?WRIO = 256/0
?DEV REG = 177400/
?INT VEC = 000220/
?BUS REQ = 5/
ENTER STMNT'S
>0010   LOAD COM0 WITH 0
>0020   READ D2048 INTO FREE
>0030   LOAD COM0 WITH 1
>0040   LETGO
>0050   IF COM0 = 1 GO TO 40
>0060   STEPUP 4
>0070   IF CYL > 0 GO TO 20
>0080   IF HEAD > 0 GO TO 20
>0090   IF SECT > 0 GO TO 20
>0100   END
>DONE
"PGMS COMPILED/MEM REFORMATTED"
*ENTER 2 AS DISK2
?ASGN DEV : RK11, 1
?RDIO = 256/0
?WRIO = 256/0
?DEV REG = 177400/
?INT VEC = 000220/
?BUS REQ = 5/
ENTER STMNT'S
>0010   IF COM0=1 GO TO 40
>0020   LETGO
>0030   GO TO 10
>0040   WRITE D2048 FROM FREE
>0050   STEPUP 4
>0060   LOAD COM0 WITH 0
```

```
>0070  IF CYL > 0 GO TO 20
>0080  IF HEAD > 0 GO TO 20
>0090  IF SECT > 0 GO TO 20
>0100  END
>DONE
"PGMS COMPILED/MEM REFORMATTED"
*ENTER 3 AS MGTAPE
?ASGN DEV : TM02, 0
?RDIO = 256/1000
?WRIO = 256/1000
?DEV REG = 172440/
?INT VEC = 000224/
?BUS REQ = 5/
ENTER STMNT'S
>0010  LOAD TM04 WITH D100
>0020  FILL D1000 AT WRIO WITH PAT 9
>0030  WRITE D1000
>0040  DECR TM04
>0050  IF TM04 > 0 GO TO 30
>0060  REWIND
>0070  LOAD TM05 WITH D100
>0080  READ D1000
>0090  VERIFY D1000 AT RDIO WITH WRIO
>0100  DECR TM05
>0110  IF TM05 > 0 GO TO 80
>0120  REWIND
>0130  END
>DONE
"PGMS COMPILED/MEM REFORMATTED"
*ENTER 4 AS PRINTER
?ASGN DEV : LP11
?RDIO = 256/132
?WRIO = 256/132
?DEV REG = 177514/
?INT VEC = 000200/
?BUS REQ = 4/
ENTER STMNT'S
>0010  LOAD TM00 WITH D96
>0020  FILL D132 AT RDIO WITH 401
>0030  FILL D132 AT WRIO WITH 20040
>0040  WRITE D132 FROM WRIO
>0050  ADD D132 AT RDIO TO WRIO
>0060  DECR TM00
>0070  IF TM00 > 0 GO TO 40
>0080  EOT
>0090  END
>DONE
"PGMS COMPILED/MEM REFORMATTED"
*RUN 1, 2, 3, 4
```

ANEXA D

Switch-uri de operare

Valoarea prestabilită este : 100000

Semnificaţia biţilor de control :

15 0 — continuă execuţia la eroare ;
   1 — oprește execuţia programului.

14 0 — dă controlul echipamentului următor, la sfîrşitul execuţiei;
1 — reia execuţia pentru aceeaşi unitate.

13 0 — tipăreşte mesaj eroare;
1 — inhibă tipărirea.

12 x — nefolosit.

11 x — nefolosit.

10 0 — oprire program la terminarea listei de echipamente;
1 — ciclează pe lista de echipamente.

9 0 — continuă execuţia pe acelaşi echipament în caz de eroare;
1 — reia execuţia pentru echipamentul următor, la eroare (bitul 9 este efectiv doar dacă bitul 15 are valoarea 0).

8 0 — verificare erori;
1 — fără verificare erori.

7 0 — nu sînt operaţii speciale;
1 — operaţii speciale.

6 x — nefolosit.

5 0 — activează timeout de I/E;
1 — fără timeout de I/E.

4 0 — afişarea automată a informaţiilor statistice conform valorii bitului 3 din OPSW;
1 — nu se afişează inf. statistice.

3 0 — afişează inf. statistice la sfîrşitul fiecărui pas;
1 — afişează inf. statistice numai după pasul final.

2 0 — şterge zona de inf. statistice după fiecare pas, exceptînd pasul final;
1 — nu se iniţializează zona de informaţii statistice decît la începutul execuţiei

1 0 — tipăreşte toate diferenţele rezultate în urma instrucţiunii VERIFY;
1 — tipăreşte doar prima diferenţă detectată de instrucţiunea VERIFY.

0 0 — tipăreşte mesajul "PROGRAM COMPLETED" după pasul final.
1 — nu tipăreşte mesajul de terminare a execuţiei programului.

ANEXA E

Operanzi predefiniţi

PAT0 100000 — walking ones
040000
020000, etc.

PAT1 077777 — walking zeroes
137777
157777, etc

PAT2 100000 — expanding ones
140000
160000, etc.

PAT3 077777 — expanding zeroes
037777
017777, etc.

PAT4 125252 — alternate ones (horiz.)
125252
125252, etc.

PAT5 052525 — alternate zeroes (horiz.)
052525
052525, etc.

PAT6 177777 — alternate ones (vert.)
000000
177777
000000, etc.

PAT7 000000 — alternate zeroes (vert.)
177777
000000
177777, etc.

PAT8 070707 — octal checkerboard
107070
070707
107070, etc.

PAT9 125252 — binary checkerboard
052525
125252
052525, etc.

PATA 000000 — count words
000001
000002, etc.

PATB 165555 — RP04 disk serial data
133333
155555
133333, etc.

II. UTILIZAREA MONITORULUI XXDP PENTRU SERVICE MINICALCULATOARE

1. MONITORUL DE TESTE XXDP

1.1. Generalităţi

Pentru manipularea programelor de test sau a diverselor fişiere în format specific pe suporturi magnetice, există programe de tip monitor de teste, specifice fiecărui tip de periferic.

Denumirea generică a acestor monitoare este XXDP (există şi XXDP + în diverse variante), unde XX are următoarele semnificaţii:

XX=**RK** (pentru disc cartridge DK)
XX=**RX** (pentru disc floppy simplă densitate DX)
XX=**RP** (pentru disc de masă DP)
XX=**TH, TM** (pentru banda magnetică MM).

Dacă acest program este salvat în mod specific (vezi 2.4/4) la începutul fizic al suportului magnetic respectiv, el este încărcabil şi executabil printr-o procedură de bootstrapare a perifericului.

1.2. Comenzile monitorului

Indiferent de perifericul de pe care este încărcat, monitorul XXDP are următoarele comenzi:

D (CR) — tipăreşte la consolă lista programelor („directory") de pe suportul magnetic;

D/L (CR) — tipărire la imprimantă a listei programelor de pe suportul magnetic ;

R_NAME (CR) — încărcarea de pe suportul magnetic a programului NAME și autostartarea acestui fișier de la adresa de memorie 200_8 ;

L_NAME (CR) — încărcarea de pe suportul magnetic a programului NAME în memorie ;

S (CR) — lansarea (startarea) programului încărcat anterior în memorie, de la adresa implicită 200_8 ;

S_ADDR (CR) — lansarea (startarea) programului încărcat anterior în memorie, de la adresa ADDR (introdusă în octal) ;

C_NAME (CR) — încărcarea și lansarea în execuție a lanțului de programe NAME. Acest lanț este de fapt un fișier de comenzi indirecte pentru monitorul XXDP (vezi 2.4/9) ;

C_NAME/QV (CR) — încărcarea și lansarea în execuție a lanțului de programe NAME. Execuția se face numai cîte o dată pentru fiecare program din lanț.

OBSERVAȚII :

a. Programele (fișierele exploatabile cu monitorul XXDP au următoarul indicativ :

NAME.EXT, unde
NAME = numele programului
EXT = extensia (tipul) programului

Pentru EXT se folosesc următoarele coduri mai frecvent :

EXT = **BIN** — fișier binar (imagine memorie executabilă)
= **BIC** — fișier binar inlănțuibil (în lanț de teste)
= **TXT** — fișier text ASCII
= **LST** — fișier listă
= **CCC** — fișier lanț de programe

Se pot declara și alte extensii (de ex. MAN pentru manual, CMD pentru fișier de comenzi etc.)

b. Programele (fișierele) exploatabile cu comanda R NAME trebuie să fie binare (BIN, BIC)
c. Programele (fișierele) executabile în lanț trebuie să aibă extensia BIC.

1.3. Lansarea monitorului XXDP.

Dacă monitorul XXDP se află pe un suport magnetic, în formă bootstrapabilă, procedura este următoarea :

1. Se lansează emulatorul de consolă ;
2. Se bootstrapează perifericul pe care avem suportul magnetic cu programele de test și XXDP ;
3. În urma bootstrapării se tipăresc la consolă cîteva mesaje ajutătoare (listarea comenzilor XXDP) după care se obține prompter-ul XXDP „.". Din acest moment se pot introduce comenzile XXDP necesare utilizatorului.
   OBS. Se poate inhiba tipărirea mesajului ajutător pentru obținerea mai rapidă a prompter-ului, cu ajutorul CTRL-C la consolă, după comanda de bootstrap-are.

1.4. Lansarea programelor în XXDP.

1. Se lansează monitorul XXDP
2. Se lansează programul NAME dorit prin una din metodele :
   a. cu comanda R_NAME (CR)
   b. cu comenzile L_NAME (CR) și apoi cu
      b.1. S (CR) start de la adresa 200_8
      b.2. S_ADDR (CR) start de la adresa $ADDR_8$
      b.3. comenzi de panou sau emulator de consolă, după ieșirea din monitorul XXDP (procedură) folosită de obicei pentru a încărca SWR înainte de startarea testului).

2. Programul utilitar UPD

2.1. Generalități programe utilitare

Există cîteva programe utilitare, cu ajutorul cărora se pot manipula programe în format XXDP.

UPD1, **UPD2**, **UPD3** — Programe utilitare generale

XTECO — Program editor de texte

COPY1, **COPY2** — Programe de copiere între periferice identice

Cele mai folosite sînt UPD2/UPD3, variante îmbunătățite ale UPD1. Pentru informații despre celelalte utilitare se listează HELP.TXT (vezi 2.4/10) care e fișierul XXDP USER MANUAL M-11-DZQXA.

2.2. UPD2 – Comenzi și periferice suportate

Lista comenzilor UPD2 cuprinde multe comenzi, din care se descriu cele mai folosite

BOOT_DEV : (CR) — bootstrapează perifericul DEV (la fel ca în emulatorul de consolă) ;

DIR_DEV : (CR) — listează la consolă cuprinsul suportului de programe de pe perifericul DEV (la fel ca D din XXDP) ;

DIRLP_DEV : (CR) — listează la imprimantă cuprinsul suportului magnetic de pe DEV ;

ZERO_DEV : (CR) — inițializează suportul de pe DEV ;

TYPE_DEV : NAME.EXT (CR) — tipărește de la consolă conținutul fișierului NAME. EXT din suportul de pe DEV ;

PRINT_DEV : NAME.EXT (CR) — tipărește la imprimantă conținutul fișierului NAME.EXT din suportul de pe DEV ;

LOAD_DEV : NAME.EXT (CR) — încarcă de pe DEV, în memorie, în format absolut (BIN, BIC) programul NAME.EXT, făcînd verificarea CHECKSUM ;

DUMP_DEV : NAME.EXT (CR) — încarcă pe DEV, cu numele NAME.EXT, conținutul memoriei în format absolut ;

PIP_DEV0 : NAME0.EXT0= DEV1 : NAME1.EXT1 (CR) — copiază de pe DEV1 fișierul NAME1.EXT1, pe DEV0 cu numele NAME0. EXT0, fără a verifica CHECKSUM ;

FILEF_DEV0 : = DEV1 : NAME.EXT (CR) — copiază rapid de pe DEV1 pe DEV0, păstrînd numele fișierului ;

FILED_DEV : NAME.EXT (CR) — șterge pe DEV, fișierul NAME.EXT ;

REN_DEV : NAME2.EXT2 = DEV : NAME1.EXT1 (CR) — schimbă pe DEV, numele fișierului NAME1 în NAME2, și extensiile respective ;

CORE (CR) — tipărește limitele memoriei afectate de transferul în memorie anterior ;

LOCORE_ADDR (CR) — fixează limita inferioară de memorie pentru un ulterior DUMP ;

HICORE_ADDR (CR) — fixează limita superioară de memorie pentru un ulterior DUMP ;

MOD_ADDR (CR) — examinează conținutul locației de memorie ADDR, cu posibilitatea modificării ei ;

DO_DEV : NAME.EXT (CR) — execută un fișier NAME. EXT de comenzi UPD2 ;

AVSE_DEV : NAME.SAV (CR) — creează un bloc autoîncărcabil pe DEV (de tip bandă magnetică) cu numele NAME ;

SAVM_DEV : (CR) — creează un bloc autoîncărcabil pe DEV (de tip disc magnetic), cu numele din operaţia de LOAD anterioară ;

TEXT DEV : NAME.EXT (CR) — creează pe DEV un fişier listing sau de comenzi (XXDP, UPD) ;

∧C (CTRL-C) — termină comanda în curs, cu revenire în UPD2 ;

∧Z (CTRL-Z) — termină comanda TEXT.

Pentru unele comenzi (PIP, FILEF) comenzile acceptă caracterul „*" în diverse combinaţii, cu semnificaţia :

*.BIN = toate fişierele cu extensia BIN ;
NAME.* = toate fişierele cu numele NAME, cu toate extensiile existente ;
. = toate fişierele cu toate extensiile ;
NA*.BIN = toate fişierele cu numele incepînd cu NA, cu extensia BIN

Perifericele suportate de UPD2, cele mai frecvent folosite, sînt :

PR = lector bandă perforată
PP = perforator bandă hîrtie
DKN = discul cartridge N (N=0÷3)
DXN = discul floppy N (N=0÷1)
DPN = discul de masă N (N=0÷1)
MMN = unitatea de bandă N (N=0÷3)

Aceste mnemonice sînt folosite în cîmpul DEV din comandă. În orice caz care conţine unitatea N, aceasta trebuie indicată, chiar dacă N=0 (nu este considerată implicit unitatea zero în UPD2).

2.3. Lansarea utilitarului UPD2

1. Se lansează monitorul XXDP.
2. Se dă comanda R_UPD2 (CR).
3. Programul îşi tipăreşte numele şi apoi cere data, care trebuie scrisă în format DD—MMM—YY (CR).

   DD — ziua
   MMM — luna (primele trei litere din numele în engleză)
   YY — anul (de ex. 85).

   După care este tipărită adresa de restart (utilă pentru relansarea directă a UPD2 încărcat în memorie) şi în final se obţine promter-ul „*" pentru UPD2, aşteptînd linii de comandă.

2.4. Exemple de folosire UPD2

Exemplele următoare sînt liniile de comenzi folosite cel mai frecvent în diferitele cazuri de lucru cu XXDP, programele de test sau utilitarul UPD2 :

1. Bootstraparea unui periferic (DEV)

   BOOT_DEV: (CR) — pentru periferice cu suporturi magnetice

2. Listarea cuprinsului unui suport de pe perifericul magnetic DEV

   DIR_DEV : (CR)

3. Ştergerea fişierelor de pe un periferic magnetic DEV

   FILED_DEV : NAME.EXT (CR)

4. Crearea unui suport magnetic autoîncărcabil pe DEV2

   — presupunem că programul XXDP specific perifericului pe care vrem să creăm acest suport bootstrapabil se află pe un suport magnetic pe DEV1

a. pentru benzi magnetice

ZERO_DEV2 : (CR)
LOAD_DEV1 : THDP.BIN (CR)
SAVE_DEV2 : THDP.SAV (CR)
LOAD_DEV1 : TMDP.BIN (CR)
SAVE_DEV2 : TMDP.SAV (CR)

unde DEV2 este MM0÷3

b. pentru discuri magnetice

ZERO_DEV2 : (CR)
LOAD_DEV1 : XXDP.BIN (CR)
SAVM_DEV2 : (CR)

unde DEV2=DXN/DKN/DPN, iar

XXDP=RXDP/RKDP/RPDP, corespunzător lui DEV2.

5. Copierea fișierelor de pe un suport magnetic (pe DEV1) pe un alt suport magnetic (pe DEV2)

a. fișier cu fișier, cu verificarea CHECKSUM și actualizarea datei cu cea introdusă la lansarea UPD2

LOAD_DEV1 : NAME1.EXT1 (CR)
DUMP_DEV2 : NAME2.EXT2 (CR)

b. fișier cu fișier, fără verificarea CHECKSUM

PIP_DEV2 : NAME2.EXT2=DEV1 : NAME1.EXT1 (CR)

c) mai multe fișiere, cu o singură comandă

c.1. PIP_DEV2 : *.BIN=DEV1 : *.BIN (CR) toate fișierele cu extensia BIN
c.2. PIP_DEV2 : NA *.BIN=DEV1 : Na*.BIN (CR) — toate fișierele cu nume începînd cu NA, și extensie BIN
c.3. PIP_DEV2 : *.*=DEV1 : *.* (CR) — toate fișierele cu toate extensiile
c.4. FILEF_DEV2 : =DEV1 : *.BIN (CR) — toate fișierele cu extensia BIN, copiere rapidă
c.5. FILEF_DEV2 : =DEV1 : NA*.BIN (CR) — toate fișierele cu nume începînd cu NA, extensie BIN, copie rapidă ;
c.6. FILEF_DEV2 : =DEV1 : *.* (CR) — toate fișierele cu toate extensiile, copiere rapidă.

Observații : cu cazurile a și b se pot schimba la transfer numele și extensia ; în toate cazurile se păstrează numele și extensia.

6. Încărcarea unui fișier de pe bandă perforată, scris în format recunoscut de ABSOLUTE LOADER

a. încărcare în memorie

LOAD PR : (CR)

b. încărcare pe un suport magnetic DEV

b.1. LOAD_PR : (CR)
DUMP_DEV : NAME.EXT (CR)
b.2. PIP_DEV : NAME.EXT=PR : (CR)

Se poate folosi b.1. dsau b.2., fișierul căpătînd numele NAME.EXT.

7. Scoaterea fișierelor pe bandă perforată (în format recunoscut de ASBOLUTE LOADER)

a. duplicare bandă perforată

PIP_PP : =PR : (CR)

b. perforare fișier de pe un suport magnetic DEV

b.1. LOAD_DEV : NAME.EXT (CR)
DUMP_PP : (CR)
b.2. PIP_PP : =DEV : NAME.EXT (CR)

8. Crearea unui fișier (program) și salvarea lui pe un suport magnetic aflat pe DEV

a. Se intră în UPD2, ținînd minte adresa de restart
b. Se iese din UPD2, intrînd în emulatorul de consolă
c. Se introduce programul în memorie cu emulatorul de consolă sau cu funcțiile de panou

d. Se reintră (**restartare**) în UPD2 de la adresa de restart

LOCORE_**INF** (CR) | Se introduc limitele de memorie ale unei zone în care am
HICORE_**SUP** (CR) | introdus programul
DUMP_**DEV** : **NAME**.EXT (CR) se salvează pe suportul magnetic cu numele dorit

Observaţii : 1) **Presupunem** un program care nu afectează zona de memorie la care am introdus **UPD2**;

2) De **obicei programele** se scriu la zone mici de memorie (peste 1 000_8) ;
3) **În cazul în care** dorim posibilitatea de lansare automată cu XXDP, programul **trebuie să aibă** prima instrucţiune la adresa 200_8;
4) **Programul se poate** perfora şi păstra pe bandă perforată, conform procedurii de **mai sus**.

9. Editarea de **fişiere listing**, fişiere text, fişiere de comenzi.
a. Se intră în **programul** utilitar UPD2
b. Se dă **comanda** TEXT_DEV : NAME.EXT (CR), după care programul tipăreşte mesajul :

MAKE **OUTPUT** READY.TYPE (CR) WHEN READY.

După ce **perifericul pe** care se editează textul (DEV) este Ready, se apasă (CR), programul **tipărind prompter-ul** " * ", aşteptînd introducerea textului.

c.1. Editarea **unui fişier** listing (text pentru tipărire)
— **Extensia este dată**, în general, de forma TXT, LST, MAN ;
— Textul **se introduce** în paginaţia dorită, trecerea la linie nouă făcîndu-se cu (CR), **obţinînd** prompter-ul pentru linia următoare ;
— **Buffer-ul de caractere** ASCII **format** în memorie este transferat pe suportul de pe **DEV** cu comanda CTRL-Z dată la sfîrşitul introducerii textului ;
— **Acest fişier se** poate lista ca la paragraful 10.

c.2. Editarea **unui fişier** de comenzi XXDP pentru execuţia unui lanţ de teste
— **Testele înlănţuibile** trebuie să aibă extensia BIC ;
— **Extensia acestui** fişier trebuie dată de forma CCC ;
— **Scrierea comenzilor** se face în forma
R_**NAME/N** (CR)
unde **N= numărul** de treceri ale testului NAME la rularea lanţului de teste ;
— **Se acceptă introducerea** de linii de comentariu, care se recunosc dacă au primul **caracter** " ; ".
— **Ieşirea din comanda** TEXT se face cu CTRL-Z, iar listarea unui fişier de comenzi **XXDP se** face ca la paragraful 10 ;
— Rularea **acestui** fişier se face cu comenzile
C_**NAME (CR)** sau
C_**NAME/QV** (CR) explicate anterior

c.3. Editarea **unui fişier** de comenzi UPD2.
— **EXT poate fi** de exemplu de forma CMD ;
— Se **pot introduce** în text linii cu comenzi UPD2, şi în plus linii începînd cu **caractere speciale**, interpretate astfel :
— **începînd** cu " ; " — linie de comentariu de tipărit în timpul execuţiei fişierului **de comenzi** ;
— **începînd** cu „$" — linie ca mai sus, dar la sfîrşitul tipăririi procesorul **intră în** HALT, repornirea făcîndu-se cu CONTINUE de la panou.
— **Introducerea** liniilor de comandă şi salvarea pe suportul de pe DEV se face la **sfîrşit**, cu CTRL-Z ;
— **Execuţia unui astfel** de fişier se face cu comanda UPD2 :
DØ_**DEV** : **NAME**.EXT (CR)
— **Listarea conţinutului** fişierului se face ca la paragraful 10.

10. Listarea unui **text sau a** unui fişier de comenzi.

Se pot lista la **consolă** sau imprimantă fişiere cu extensia TXT, LST, MAN, CMD, CCC sau altă extensie **declarată la** crearea fişierului, cu comandă TEXT (din UPD2)
Comenzile sînt :

TYPE_DEV : **NAME.EXT** (CR) listare la consolă
PRINT_DEV : **NAME.EXT** (CR) listare la imprimantă

Ciclul

„Calculatoare personale și programarea lor"

FAMILIA DE CALCULATOARE PERSONALE ROMÂNEȘTI PRAE ȘI LIMBAJUL SĂU BASIC

ing. *Patrubany Nicolae*
ing. *Pop Baldi Nicolae*
mat. *Kiss Alexandru*
mat. *Socaciu Nicușor*
fiz. *Szász Detre*

I.T.C., Filiala Cluj-Napoca

0. CALCULATOARE PRAE. CARACTERISTICI FUNCȚIONALE

0.1. Introducere

Cuvîntul PRAE (a se citi pre) este un prefix latin și înseamnă un început. Numele sugerează convingerea autorilor că familia de calculatoare care poartă acest nume reprezintă un preludiu, o deschidere către un mod nou, calitativ superior, de utilizare a calculatoarelor în România.

Această familie de calculatoare a fost elaborată în întregime la Filiala din Cluj-Napoca a Institutului de Cercetări pentru Tehnică de Calcul și este produs în serie la Fabrica de Memorii Timișoara.

Prototipul PRAE (vezi figura) a fost prezentat în public la Sesiunea din Bușteni a Cercului Utilizatorilor de Microcalculatoare și Terminale Programabile din noiembrie 1983.

Familia de microcalculatoare personale PRAE este clădită pe o unitate centrală de 8 biți (microprocesorul Z 80), ea acoperind o gamă largă de aplicații, începînd cu PRAE-T care este un bun de larg consum pînă la PRAE-M un calculator profesional.

PRAE-T este destinat în primul rînd tinerilor, pentru care el poate reprezenta un excelent mijloc de aprofundare a cunoștințelor în orice domeniu. El poate fi utilizat și ca mijloc de agrement elevat. PRAE-T folosește la maximum aparatele electronice existente în gospodăriile cetățenilor :

— televizorul servește ca dispozitiv de afișare iar casetofonul audio ca memorie externă pentru programe și date. Prin orientarea sa către bunurile de larg consum, acest calculator reprezintă un prim pas către o societate în care informația devine avuție națională, o societate în care fiecare individ va fi capabil să folosească calculatorul în vederea prestării unei munci calitativ superioare.

PRAE-L este un calculator ce poate fi utilizat cu succes în laboratoare de cercetare și învățămînt precum și în atelierele de proiectare.

PRAE-M (M specifică faptul că este varianta maximă a familiei) este un microcalculator profesional competitiv cu produse ca APPLE II, COMMODORE 64, M 118 etc.

PRAE-M este dotat cu o unitate duală de disc flexibil de 5″ 1/4 și cu un sistem de operare orientat pe disc. Astfel s-a creat o ambianță de calcul în care utilizatorul dispune de majoritatea limbajelor de programare de nivel înalt.

Familia de calculatoare PRAE prezintă o compatibilitate de sus în jos astfel încît aplicațiile elaborate pe membrii inferiori ai familiei să poată fi transpuse fără modificări pe membrii superiori.

Arhitectura hardware permite transformarea membrilor inferiori ai familiei în oricare membru superior prin simpla adăugare de elemente funcționale, servindu-se astfel interesele beneficiarilor.

Ca limbaj de programare de nivel înalt comun tuturor membrilor familiei folosește PRAE-BASIC, un interpretor deosebit de performant, compatibil cu standardurile internaționale (ANSI, CAER). Precizia de calcul este de 11 cifre semnificative.

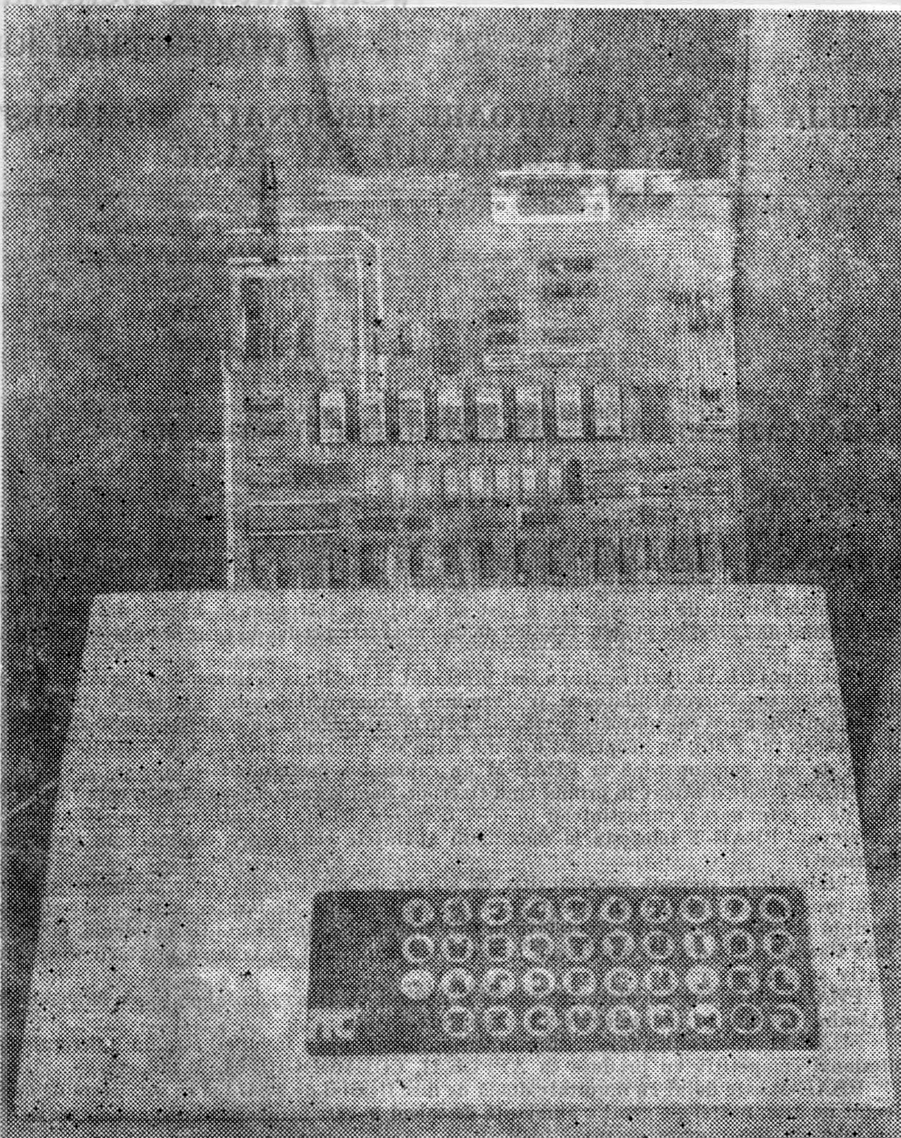


Fig. A.

Timpul de rulare pentru programul de test BM-7 acceptat internațional pentru calificarea interpretoarelor BASIC este de 53 s.

Prezenta lucrare cuprinde manualul de utilizare a componentelor software, monitorul și interpretorul BASIC V3.5 cum și anumite variabile sistem folositoare utilizatorului. Arhitectura hardware va face obiectul unei alte lucrări.

Cea mai mare parte a lucrării este dedicată învățării limbajului BASIC-PRAE.

0.2. Software rezident

În cei 16 k EPROM se află un monitor, interpretorul PRAE-BASIC 3.5, interfață tastatură, mașină de scris și casetofon, generator de caractere.

Monitorul are următoarele comenzi :

F (Fill)	FAdr1 Adr2 Valoare	(Intre Adr1 și Adr2 se umple cu valoarea Valoare)
G (Go)	GAdr1 Adr2 GAdr1 G Adr2	(Se folosește în timpul depanării unui program)
L (List)	L (CR) L (A, B, C, D, E, H, L, M, SP)	Listarea și/sau modificarea registrelor)
M (Move)	MAdr1 Adr2 Adr3	(Se mută conținutul dintre Adr1 și Adr2 începînd de la Adr3)
Q (Quit)	Q (CR)	(Salt în Basic cu distrugere program Basic)
R (Return)	R (CR)	(Return în Program Basic în locul întrerupt)
S (Substitute)	SAdr	(Substituie conținutul memoriei)

Adrese mai semnificative din monitor :

01EA	Convertește o valoare binară pe 4 biți în hexa extern Valoarea binară se află în A, caracterul imprimabil se returnează în C. Nici un alt registru afectat.
01F2	Retur de car. Regiștri afectați : A, C.
0241	Convertește o valoare binară între 0—15 în hexa extern și imprimă. Valoarea se află în A. Regiștri afectați A, C.
024C	Convertește o valoare binară între 0—1255 în hexa extern și imprimă. Valoarea în A, registru C afectat.
0247	Convertește o valoare binară între 0—65535 în format hexa extern și îl imprimă.
2D5	Intrare de la consolă ; caracter recepționat în registrul A.
1F9	Ieșire la consolă ; caracter de transmis în registrul C.
1E5	Tipărire spațiu ; BLK
25C	Decodifică un caracter hexa extern în valoarea lui binară ; intrare în registrul A ; în cazul unui caracter eronat $C_y=1$.

Interpretorul BASIC primește spre interpretare programe de la tastatură sau de la perifericul extern (casetă/bandă magnetică). Calculatorul funcționează și în regim de calculator de buzunar. Se pot executa comenzi, calcule imediate, ceea ce se identifică prin faptul că linia respectivă nu conține număr de ordine.

În cazul cînd o linie conține un număr de ordine între 1—65535 ea se structurează într-un program și se lansează în execuție prin comanda RUN.

Calculele în BASIC se fac în dublă lungime. Reprezentarea internă este flotantă. Se reprezintă pe 6 octeți în care 1 octet este exponentul binar (caracteristica) și 5 octeți mantisa.

Este permis lucrul cu masive de orice dimensiuni, se pot folosi funcții definite. Există variabile de tip șiruri etc.

Biblioteca BASIC conține în afara funcțiilor matematice și funcții referitoare la operațiI cu șiruri de caractere.

PRAE-BASIC are rutine pentru grafică, foarte rapide și ușor utilizabile.

Pentru a satisface nevoile celor mai diverși utilizatori amintim posibilitatea compunerii de melodii pe acest calculator avînd la dispoziție instrucțiunea BEEP.

Toate aceste posibilități oferite de BASIC vor fi detaliate la descrierea limbajului.

0.3. Software pe casetă. Extinderea interpretorului PRAE-BASIC

Pentru a nu se mări necesarul de memorie fixă (16 kocteţi) anumite funcţii ale interpretorului PRAE-BASIC se livrează pe casetă. Aceste funcţii rulează în RAM.

Amintim două din cele mai importante module:

a) IOARRAY — permite salvarea şi recitirea de pe casetofon a variabilelor masive de tip numeric şi/ sau alfanumeric.

b) MATRIX — permite operaţii pe matrici cum ar fi: matricea inversă, matricea transpusă, determinantul unei matrici, adunarea, scăderea şi înmulţirea matricilor, iniţializarea, citirea şi imprimarea matricilor.

0.4. Adrese de subprograme utile

3151	Sistem Console Input	(Caracter în A)
3154	Sistem Console Output	(Caracter în C)
3169	Sistem Console Status	(FF sau caracterul în A)
316C	Serial OUT	
317B	Serial INPUT	
31A8	Test EPROM	
3187	Scrierea pe casetă a conţinutului de la 4080—40FF H	
318A	Citirea unui bloc (curent) de pe casetă la adresa 4080 H	
318D	Intrare tastatură proprie (reg A)	
3190	Ieşire pe ecran (reg C)	
319F	Console Status Serial	
31A2	Console Status tastatură proprie	
31A5	Desen miră (HL distrus)	
31A8	Test EPROM	
31AE	Test RAM (Revine după terminal afişarea adresei primei celule defecte).	

0.5. Celule de lucru, variabile sistem mai importante

BEG : 4004—4005 H	Adresă început ecran pentru rutinele de tipărire
BEGPLT : 401A—401B H	Adresă început ecran pentru rutinele de grafică
FIN : 4008—4009 H	Adresă sfîrşit ecran, coincide cu adresa sfîrşit RAM.
PNT : 4006—4007 H	Adresă curentă (pontor) de afişare text pe ecran.
FONDLI : 400AH	Starea SHIFT S tastatură
TIPCON : 400BH	Tip consolă (tastatură proprie sau serială)
SWITCH : 400CH	Starea după instrucţiunea SWITCH Basic (Valoare 0, 1 sau 2)
ULTCAR : 4019H	Ultimul caracter tastat în timpul execuţiei unui program BASIC (simulare INKEY).
STARES : 4020 H	Validarea sunetului la tastare (=0 — tastatură fără sunet =0 — tastatură cu sunet)
UPAD : 4010 H	Interfaţa serială
+0	Starea curentă a portului de ieşire
+1	Adresă port ieşire (80 H)
+2	Număr biţi pe unitate de transfer (5, 6, 7, sau 8)
+3	Viteza (254_{10} pentru 300 baud 127_{10} pentru 600 baud 64_{10} pentru 1200 baud etc.)
+4	Paritatea $b_0=0$ transfer fără paritate $b_0=1$ transfer cu paritate $b_1=0$ paritate pară $b_1=1$ paritate impară $b_6=$ bitul de paritate rezultat în cazul transferului cu paritate
+5	Număr biţi STOP (1 sau 2)
DENSIT : 4018 H	Densitate de înregistrare pe casetp (10_{10} stand, DENSIT $\geq$ 6)
LONG : 403C—403D	Lungime bloc casetă (LONG $\leq 134_{10}$)

1. INTRODUCERE ÎN BASIC

BASIC-ul este un limbaj de programare conversaţional folosind cuvinte cheie sugestive din limba engleză şi notaţiile matematice familiare, ceea ce îl recomandă ca unul din limbajele cele mai uşor asimilabile.

În acelaşi timp, pentru programatorii experimentaţi, limbajul pune la dispoziţie tehnici avansate pentru rezolvarea eficientă a problemelor de orice natură.

1.1. Convenţii de notaţii

În cuprinsul acestui manual sînt utilizate unele convenţii de notaţii pentru explicarea detaliată a sintaxei instrucţiunilor şi a elementelor compuse ale limbajului.

Pentru descrierea sintaxei instrucţiunilor se folosesc următoarele notaţii :

1. Elementele scrise cu litere mari (ex. LET, PRINT, IF etc.) trebuie să apară în instrucţiuni aşa cum sînt notate, ele constituind vocabularul limbajului (cuvinte cheie) ;

2. Elementele scrise cu litere mici şi închise între paranteze unghiulare indică elementele sintactice ale unei instrucţiuni sau comenzi.

Exemplu :

[LET] <variabilă> = <expresie>

3. Parantezele drepte indică posibilitatea prezenţei sau absenţei elementului cuprins între ele (element opţional) ;

4. Parantezele acolade indică posibilitatea alegerii uneia din variantele indicate, separate prin semnul "/".

Exemplu :

IF <expresie> THEN {<instr.> / <nr. linie>}
[ELSE <instr.> / <nr. linie>}]

1.2. Programul BASIC-PRAE

Un program constă din una sau mai multe „linii program". Din punct de vedere fizic, linia program este echivalentă cu linia terminal utilizator. „Linia program" este definită ca avîd structura

<nr. linie> <lista de instrucţiuni>

Fiecare linie program începe cu un număr de linie care va identifica linia în contextul întregului program şi indică ordinea de execuţie a instrucţiunilor. O linie program poate conţine una sau mai multe instrucţiuni separate prin simbolul ':'. Lista de instrucţiuni a unei linii poate fi şi vidă.

Fizic, linia program reprezintă totalitatea caracterelor cuprinse între promterul emis de sistem şi terminatorul unei linii terminal (CR sau "RETUR DE CAR").

Lista de instrucţiuni are forma :

<instr.> : <instr.> : <instr.> ...

Fiecare instrucţiune începe cu un cuvînt cheie exprimat în limba engleză care indică tipul operaţiei implicate de respectiva instrucţiune.

Capitolul 3 conţine descrierea tuturor instrucţionilor BASIC-PRAE.

1.3. Număr de linie

Fiecare linie program BASIC-PRAE începe cu un număr de linie.

Numărul de linie :

— indică ordinea de execuţie a instrucţiunilor programului ;

— permite schimbarea ordinii normale de execuţie prin instrucţiuni de salt şi cicluri ;

— permite punerea la punct a unui program dînd posibilitatea schimbării oricărei linii fără afectarea restului programului ;

— permite indicarea instrucţiunii eronate în timpul execuţiei programului.

Numărul de linie este un număr întreg format din 1 pînă la 5 cifre zecimale putînd avea valoarea cuprinsă între 1 şi 65 529.

Programul se execută în ordinea crescătoare a numerelor de linie (ordine logică).

Remarcă : la prima scriere a unui program este recomandată numerotarea liniilor folosind raţia 5 sau 10 pentru a permite, în faza de punere la punct a programului, inserarea unor instrucţiuni între instrucţiunile existente ale programului.

Introducerea unei linii avînd un număr de linie deja existent în program, provoacă ştergerea liniei vechi şi înlocuirea sa prin linia nou introdusă.

Introducerea unei linii în care numărul de linie este imediat urmat de caracterul CR (RETUR DE CAR) produce ştergerea liniei cu acelaşi număr de linie dacă aceasta a existat.

1.4. Linie simplă. Linie multiplă

Instrucţiune multiplă

O linie program poate conţine una (linie simplă) sau mai multe (linie multiplă) instrucţiuni separate între ele prin caracterul ':'.

Remarcă : promterul componentei BASIC care se imprimă la începutul unei linii terminal în aşteptarea introducerii unei noi linii terminal este simbolul '—'.

Exemplu :

```
10 LET A=0.1
50 FOR I=1 TO 5 : PRINT I ^2 : NEXT I
```

Remarcă : transferul controlului poate fi executat numai la prima instrucţiune a unei linii multiple.

Orice program poate fi salvat, listat, modificat sau executat prin comenzile proprii componentei.

1.5. Utilizarea caracterelor ' ' (spaţiu) şi TAB (CTRL I)

Caracterul ' ' (spaţiu) poate fi utilizat singur sau în mod repetat oriunde în textul unei instrucţiuni pentru a servi estetica textului sursă.

De exemplu linia :

```
100 FORI = 1TO10 : PRINTI 2 : NEXTI
```

poate fi scrisă :

```
100 FOR I = 1 TO 10 : PRINT I ^2 : NEXT I
```

mărind evident posibilitatea remarcării rapide a elementelor componente ale liniei.

Caracterul TAB poate fi deasemeni folosit oriunde în textul unei instrucţiuni pentru a fi uşor citit. Textul liniei în care se foloseşte caracterul TAB apare identic la listare cu textul introdus.

Exemplu de utilizare :

```
10 FOR X=1 T0 10
20 FOR Y= T0 10
30 A(X, Y)=1/(X+Y—1)+A(X, Y)
40 NEXT Y
50 NEXT X
```

2. ELEMENTELE LIMBAJULUI BASIC-PRAE

2.1. Principii de realizare

Alfabetul limbajului BASIC-PRAE conţine :

a) litere : A, B, ..., Z
b) cifre : 1, 2, ..., 9, 0
c) caractere speciale

În setul caracterelor speciale intră toate caracterele care posedă un cod imprimabil.

Pornind de la caracterele de bază ale limbajului, folosind reguli sintactice precise (detaliat descrise la timpul potrivit) se obţin elementele compuse ale limbajului.

2.2. Elemente aritmetice

2.2.1. *Constante numerice*

În limbajul BASIC-PRAE toate numerele sînt tratate ca numerele zecimale conform notaţiei matematice uzuale.

Pentru numere este folosită denumirea de constantă numerică, pornind de la faptul că acestea pentru programul în sine au tot timpul aceeaşi valoare.

O constantă numerică este un şir de cifre zecimale a cărui valoare reprezintă o valoare numerică (în sens obişnuit matematic). Valoarea constantelor numerice acceptate de limbaj trebuie să fie cuprinsă în intervalul real (10^—38, 10^+38).

În limbaj se recunosc trei formate de exprimare a constantelor numerice:

— format întreg 123

— format real 123.456

— format exponenţial (cu puterile lui 10). 12.3E6

Constantele pot fi reprezentate în oricare din cele trei formate.

Sistemul BASIC consideră constantele ca fiind reale reprezentate în format flotant.

Dacă constantele sînt scrise cu caracterul & în faţa, ele sînt constante hexazecimale (&04AC).

2.2.2. *Variabile simple*

Variabila este un simbol algebric reprezentînd un număr.

Numele de variabilă este format dintr-o literă urmată de oricîte caractere alfanumerice, dintre care numai primele două sînt luate în considerare.

Valoarea variabilei este reprezentată pe 6 octeţi în flotant.

De exemplu:

CORECT	INCORECT
I	2A
A8	2B
X	12

Unei variabile I se poate atribui o valoare prin:

— instrucţiunea LET;

— instrucţiunea INPUT sau READ.

Valoarea unei variabile rămîne constantă pe intervalul cuprins între două atribuiri.

În momentul lansării în execuţie a unui program, toate variabilele acestuia sînt puse la 0.

2.2.3. *Variabile indexate*

Pentru prelucrarea listelor, tabelelor sau matricilor limbajul posedă un element special numit „variabilă indexată". BASIC-PRAE permite prelucrarea variabilelor indexate avînd unul, doi sau pînă la 255 indici.

Numele unei variabile indexate este format dintr-un nume admis de variabila simplă urmat de una sau două expresii de indici în paranteze.

Exemplu:

A(I), B(I, J), C (I+K, J+K)

Astfel o listă de 6 elemente căreia i se atribuie numele A poate fi reprezentată prin variabila indexată A (I) unde I poate lua valorile 0, 1, 2, ..., 5.

A(0), A(1), A(2), A(3), A(4), A(5)

Prin această convenţie fiecare element al listei poate fi reprezentat printr-o variabilă indexată avînd indicele egal cu ordinul elementului în listă. O matrice (tablou) bidimensională cu numele B se poate defini şi reprezenta grafic astfel :

B (0, 0), B (0, 1), . . ., B (0, N)
B (1, 0), B (1, 1), . . ., B (1, N)
B (2, 0), B (2 ,1), . . ., B (2, N)
B (M, 0), B (M, 1), . . ., B (M, N)

Limitele între care pot varia indicii unei variabile indexate pot fi declarate într-un program :

— implicit, folosind prima reprezentare legală a variabilei indexate ;
— explicit, prin instrucţiunea de declarare DIM.

Într-un program este posibilă definirea prin acelaşi nume a unei variabile simple şi a unei variabile indexate.

Exemplu :

A — variabilă simplă reală
A(X) — variabilă indexată reală

2.2.4. *Expresii numerice*

Expresia numerică este definită ca un grup de simboluri, elemente numerice ale limbajului, care poate fi evaluat. Expresiile numerice sînt compuse din constante numerice, variabile simple, variabile indexate sau combinaţii ale acestora separate prin operatori aritmetici, operatori de relaţie sau operatori logici.

2.2.4.1. *Expresii aritmetice*

Expresia aritmetică este compusă din elemente numerice (constante, variabile simple sau indexate), separate între ele prin operatori aritmetici.

Reprezentarea unei expresii aritmetice este similară notaţiei matematice uzuale.

Exemplu :

A+0·5*B(I, J)/2

Operatorii aritmetici admişi sînt :

Operator	Exemplu	Semnificaţie
+	A+B	adună A cu B
—	A—B	scade B din A
*	A×B	înmulţeşte A cu B
/	A/B	împarte A prin B
^	A^B	calculează A la puterea B
+	+A	Operaţia unară +
—	—A	Operaţie unară —

Dacă într-o expresie aritmetică apar mai mulţi operatori, evaluarea expresiei se va face conform priorităţii matematice uzuale a acestor operatori.

Într-o expresie oarecare data, ordinea de execuţie (prioritatea) este următoarea :

1) evaluarea expresiilor cuprinse între paranteze
2) în absenţa parantezelor ordinea este :
   a) operatori unari
   b) ridicarea la putere
   c) înmulţirea şi împărţirea
   d) adunarea şi scăderea
3) între operatori cu acelaşi nivel de prioritate, relaţia de ordine este indicată prin regula de la stînga la dreapta.

De exemplu în expresia:

A=B*((C ^2+4)/X)

Ordinea de evaluare este:

P1—C ^2
P2 — rezultat P1+4
P3 — rezultat P2/X
P4 — rezultat P3*B
P5 — rezultat P4 se atribuie lui A

2.2.4.2. *Expresii de relaţie*

Expresia de relaţie definește prin intermediul operatorilor de relaţie legătura între două entităţi.

În forma cea mai simplă, expresia de relaţie este compusă din două expresii aritmetice separate între ele printr-un operator de relaţie.

Operatorii de relaţie admiși sînt:

Simbol matematic	Simbol BASIC-PRAE	Exemplu	Semnificaţie
=	=	A=B	A egal cu B
<	<	A<B	A mai mic decît B
>	>	A>B	A mai mare decît B
<=	<=	A<=B	A mai mic/egal cu B
>=	>=	A>=B	A mai mare/egal cu B
≠	<>	A<>B	A diferit de B

Evaluarea expresiei de relaţie se execută prin:

— evaluarea expresiilor aritmetice componente,
— evaluarea relaţiei dintre valorile obţinute.

Valoarea unei expresii de relaţie este o valoare logică de adevăr („ADEVĂRAT" sau „FALS").

Expresia de relaţie redusă la o expresie aritmetică exprimă în fapt relaţia dintre valoarea expresiei aritmetice și valoarea 0, și are valoarea însăși a expresiei aritmetice.

2.2.4.3. *Expresia logică*

Expresia logică este compusă din una sau mai multe expresii de relaţie separate între ele prin operatori logici.

Operatorii logici simbolizează operaţiile logice din logica matematică obișnuită. Operatorii logici admiși sînt:

Simbol	*Exemplu*	*Semnificaţie*
NOT	NOT L	operaţia logică „NOT"
AND	L1 AND L2	operaţia logică „SI"
YOR	L1 OR L2	operaţie logică „SAU"

Tabelele de adevăr ale operaţiilor sînt cele din logica matematică:

L	NOT L
A	F
F	A

L1	L2	L1 AND L2
A	A	A
A	F	F
F	A	F
F	F	F

L1	L2	L1 OR L2
A	A	A
A	F	A
F	A	A
F	F	F

Operaţia se execută bit cu bit dacă operanzii sînt în intervalul 0, 65535, altfel funcţia este refuzată cu mesajul de eroare corespunzător.

Ordinea de evaluare a unei expresii logice este:

— evaluarea expresiilor de relaţie,

— evaluarea operaţiilor logice în ordinea :

NOT
AND
OR

— între operaţii cu acelaşi nivel ierarhic evaluarea se face de la stînga la dreapta.

2.2.5. *Funcţii matematice standard*

Multe din funcţiile matematice comune (sinus, funcţia putere, funcţia logaritmică) pot fi executate în sistemul BASIC prin intermediul unor funcţii aritmetice speciale, recunoscute în limbajul BASIC-PRAE.

De exemplu, pentru calcularea valorii sin (75 de grade) se creează o expresie simbolică formată din numele simbolic al funcţiei respective (SIN) urmat de valoarea parametrului în radiani închis între paranteze :

SIN (75*PI/180)

Este prezentat în continuare tabelul funcţiilor matematice recunoscute ca elemente ale limbajului BASIC-PRAE :

Notaţie	Simbol	Semnificaţie matematică
$\lvert x \rvert$	ABS (X)	Valoare absolută
sign x	SGN (X)	Generează : 1 dacă x>0 —1 dacă x<0 0 dacă x=0
[x]	INT (X)	Parte întreagă din x
cos (x)	COS (X)	Cosinus de x (radiani)
sin (x)	SIN (X)	Sinus de x (radiani)
tg (x)	TAN (X)	Generează o valoare reală egală cu valoarea funcţiei tangentă pentru argumentul x dat în radiani
arctg (x)	ATN (X)	Arctangenta lui x
$\sqrt{x}$	SQR (X)	Rădăcină pătrată din x
e^x	EXP (X)	Calculează e ^x
ln x	LOG (X)	Calculează logaritmul natural din x

2.2.6. *Funcţii aritmetice definite*

În practica de programare se observă adeseori necesitatea executării unei secvenţe de instrucţiuni sau a unei expresii în mai multe puncte ale programului.

Limbajul BASIC-PRAE pune la dispoziţia utilizatorilor săi posibilitatea definirii unice a unei astfel de expresii sau secvenţe de instrucţiuni şi simplul apel al acestora în diferite puncte ale programului.

Aceasta înseamnă că limbajul permite utilizatorului să definească funcţiile sale proprii şi să apeleze apoi aceste funcţii în acelaşi mod în care se apelează funcţiile aritmetice standard.

Numele unei funcţii utilizator este format din maximum 4 caractere :

— primele două caractere standard egale cu „FN"

— un nume de variabilă corect (o literă urmată de maximum 1 caracter alfanumeric).

Exemplu :

CORECT	*INCORECT*
FNI	FN2
FNA8	FN1B

Definiţia unei funcţii utilizator se introduce printr-o instrucţiune DEFFN.

Apelul unei funcţii se realizează prin introducerea numelui de funcţie urmat de parametrul (parametrii) de apel.

La apelul unei funcții se face o corespondență strictă între numărul și tipul parametrilor formali și numărul și tipul parametrilor de apel (actuali).

Conform structurii definiției, funcțiile definite utilizator se împart în două categorii:

— funcție simplu-definită, a cărei definiție constă dintr-o expresie numerică (aritmetică, de relație sau logică)

— funcție multilinie, a cărei definiție constă dintr-o secvență de instrucțiuni cuprinse între instrucțiunile DEFFN și FNEND.

Pentru înțelegerea mai detaliată, a se vedea paragraful „Instrucțiunea DEFFN".

2.3. **Elemente nenumerice**

Limbajul BASIC-PRAE permite manipularea unor informații numerice sub forma șirurilor de caractere.

Prin șir de caractere se înțelege o secvență de caractere BASIC-PRAE admise, privită ca o unitate.

Un șir de caractere poate conține litere, cifre sau caractere speciale. Lungimea unui șir de caractere se definește ca fiind numărul de caractere BASIC-PRAE componente.

În cele ce urmează vom numi „șir vid" șirul de caractere avind lungimea 0.

Pe mulțimea șirurilor de caractere se pot defini operații ca :

— adunarea (concatenarea) a două șiruri
— extragerea unui subșir dintr-un șir dat
— modificarea elementelor componente ale unui șir etc.

Toate elementele numerice BASIC-PRAE au corespondente în setul elementelor nenumerice :

Exemplu :

constante numerice — constante șir
variabile numerice simple — variabile simple șir
masive numerice — masive șir
funcții standard numerice — funcții standard șir.

2.3.1. *Constante șir de caractere*

Prin constanta șir de caractere se înțelege orice șir închis între simbolurile "" (ghilimele).

O singură excepție de la acestă definiție există și apare în cazul constantelor șir de caractere dispuse într-o instrucțiune DATA (a se vedea paragraful „Instrucțiunea DATA").

Exemplu :

"123·45"
"LET'S BEGIN"

2.3.2. *Variabile simple de tip șir*

Variabila simplă de tip șir este un simbol reprezentînd un șir de caractere.

Numele unei variabile simple șir este format din :

— nume variabilă (litera urmată opțional de maximum 5 caractere alfanumerice)
— caracterul „$"

Exemplu :

Variabile simple numerice	*Variabile simple șir*
A	A $
B2	B2 $
M	M $

Într-un program pot exista variabile simple numerice și variabile simple șir cu același nume.

Exemple :

A, A$

Ca şi în cazul variabilelor numerice, unei variabile simple de tip şir i se poate atribui o valoare prin instrucţiunile LET, INPUT, READ.

Variabilele de tip şir sînt iniţializate la execuţia unui program cu valori egale cu „şirul vid".

2.3.3. *Variabile indexate de tip şir*

BASIC-PRAE permite definirea variabilelor indexate de tip şir pentru rezolvarea prelucrării listelor şi masivelor de şiruri de caractere.

Variabilă indexată de tip şir este un simbol care constă din :

— un nume corect de variabilă
— caracterul „$"
— unul sau pînă la 255 indici între paranteze

Exemple :

A$(5)
A1$(I, J)

De reţinut că expresiile de indici nu pot fi decît expresii aritmetice.

Limitele între care pot varia indicii unei variabile indexate de tip şir pot fi declarate în program :

— implicit prin prima referinţă
— implicit prin instrucţiunile de declarare DIM

Este posibilă utilizarea simultană a unei variabile simple şir şi a unei variabile indexate şir cu acelaşi nume.

Exemplu :

A$ şi A$(10)

De asemenea este permisă utilizarea simultană a aceluiaşi nume pentru variabilele numerice şi variabilele şir.

Exemplu :

A, A$, A(I), A$(I)

2.3.4. *Expresii nenumerice*

Expresia şir este definită ca un grup de elemente nenumerice care poate fi evaluat.

Expresiile şir sînt compuse din constante şir, variabile simple sau indexate de tip şir sau combinaţii ale acestora separate prin operatori :

— operatorul de concatenare (simbol +)
— operatori de relaţie (expresie condiţie şir)

Operatorul de concatenare marchează operaţia prin care din două şiruri date se poate obţine un nou şir prin alipirea şirurilor date.

Operatorul de concatenare se notează prin simbolul matematic „+". El operează numai asupra şirurilor şi nu este comutativ.

Exemplu :

"ABC"+"123" produce şirul "ABC123"

Operatorii de relaţie operează asupra şirurilor cu respectarea ordinii alfabetice. Compararea a două şiruri se face caracter cu caracter de la stînga la dreapta.

Operatorii de relaţie permişi şi interpretarea lor figurează în tabelul următor :

Operator	*Exemplu*	*Semnificaţie*
=	A$=B$	Şirurile A$ şi B$ sînt identice
<	A$<B$	A$ mai mic alfabetic ca B$ sau A$ subşir al lui B$
>	A$>B$	A$ mai mare alfabetic ca B$ sau B$ subşir stîng al lui A$.

Operator	*Exemplu*	*Semnificaţie*
<=	A$<=B$	A$ mai mic sau egal alfabetic cu B$
>=	A$>=B$	A$ mai mare sau egal alfabetic cu B$
≠	A$<>B$	A$ diferit de B$

Expresia de relaţie redusă la un singur element marchează în fapt relaţia între elementul şi "şir vid".

2.3.5. *Funcţii standard pe şiruri de caractere*

BASIC-PRAE recunoaşte ca şi în cazul funcţiilor matematice standard, un set de funcţii asupra şirurilor de caractere.

Apelul unei astfel de funcţii se poate face în orice punct al unui program prin indicarea numelui simbolic al funcţiei şi a parametrului sau parametrilor de apel.

Funcţiile standard pe şirurile de caractere acceptate sînt:

Cod apel	*Semnificaţie*
LEFT$(A$,N)	Extrage subşirul stîng de lungime N al şirului A$. LEFT$ ("ABCD",2) produce "AB".
RIGHT$(A$,N)	Extrage subşirul drept de lungime N din şirul A$. RIGHT$ ("ABCD",2) produce "CD".
MID$(A$,N1,L)	Extrage din şirul A$ un subşir de lungime L începînd cu al N1-lea caracter. MID$ ("ABCD",2,2) produce "BC".
LEN(A$)	Generează o valoare întreagă egală cu lungimea şirului A$. LEN ("ABCD") produce 4.
CHR$(N)	Produce un şir de lungime 1 care conţine codul ASCII de valoare N. CHR(42) produce "*".
ASC(A$)	Generează valoarea zecimală întreagă a codului ASCII a primului caracter al şirului A$. ASC("ABC") generează 65.
INSTR(A$,B$,S,L)	Generează o valoare întreagă ca rezultat al căutării subşirului B$ de lungime L în şirul A$ începînd cu caracterul al S-lea din A$. Dacă B$ nu se află în A$ se produce valoarea întreagă 0. Dacă B$ se află în A$ produce valoarea întreagă egală cu indicele caracterului din şirul A$ începînd cu care cele două şiruri coincid Instr ("ABCDEF", "X",2) produce 0 Instr ("ABCDEF", "EF",2) produce 5. Prin convenţie "şirul vid" este subşir stîng al oricărui şir.
SPC(L%)	Generează un şir de blancuri de lungime L%.
STR$(N)	Generează un şir de caractere numerice reprezentînd valoarea de editare a numărului N. STR$(1234567) produce "1234567".
VAL(A$)	Generează valoarea flotantă a şirului de caractere numerice A$. VAL(„123") generează valoarea în reprezentare a numărului 123.

2.3.6. *Funcţiile definite pe şiruri de caractere*

Limbajul BASIC-PRAE pune la dispoziţia utilizatorilor posibilitatea definirii unor funcţii proprii de tip şir de caractere.

Indicaţia de tip se marchează prin introducerea caracterului "$" imediat după numele funcţiei.

Definiţia şi apelul funcţiilor definite de tip şir respectă aceleaşi reguli ca cele de tip numeric. Funcţia definită de tip şir poate fi simplă sau multilinie. Într-un program este permisă utilizarea simultană a unei funcţii numerice şi a unei funcţii de tip şir cu acelaşi nume.

Exemplu:

FNA, FNA, FNA$

2.3.7. *Funcţii grafice*

Interpretorul BASIC-PRAE recunoaşte 6 funcţii grafice pentru a face posibilă desenarea pe ecran. Ecranul este considerat ca fiind compus din 256 linii şi 256 coloane (64*1 024 puncte).

Pentru stabilirea fondului ecranului există o directivă : SWITCH valoare

Dacă valoarea este 0, ecranul are fond alb ; dacă are valaorea 1, fondul este negru ; dacă valoare are valoarea 2 se realizează o intrare-ieşire serială.

Funcţiile grafice sînt :

Cod apel	*Semnificaţie*
CIRCLE (x, y, r)	desenează un cerc avînd centrul în punctul (x,y) de raza r
CIRCLEC (x, y, r)	şterge de pe ecran cercul cu centrul (x,) şi de rază r
PLOT (x, y)	desenează un punct pe ecran de coordonare (x,y)
PLOTC (x, y)	şterge punctul de coordonate (x,y)
DRAW (x, y)	leagă cu o dreaptă ultimul punct desenat pe ecran cu punctul de coordonate (x, y)
DRAWC (x, y)	şterge dreapta desenată
x, y, r	sînt expresii numerice, cu valori între 0 şi 255.

3. INSTRUCŢIUNI BASIC-PRAE

În acest capitol este dată o descriere detaliată a instrucţiunilor admise în limbajul BASIC-PRAE.

Sînt folosite următoarele convenţii şi notaţii :

<notaţie>	Semnificaţie
<nr. linie>	Orice număr de linie admis
<instr.>	Orice instrucţiune admisă
<expr.>	Orice expresie admisă indiferent de tip (numerică sau şir)
<expr. num.>	Orice expresie numerică admisă (aritmetica, de relaţie, logica)
<expr. şir>	Orice expresie şir admisă
<expr. aritm.>	Orice expresie aritmetică admisă
<expr. rel.>	Orice expresie de relaţie admisă
<expr. log.>	Orice expresie logică admisă
<var.>	Orice variabilă admisă indiferent de tip (simplă, incexată, întreagă, şir, reală)
<mesaj>	Lanţ de caractere admis ca şi comentariu
<cond.>	Orice expresie logică admisă, care se poate reduce la o expresie de relaţie sau aritmetică
<const.>	Orice constantă numerică sau şir admisă
<şir>	Orice constantă şir admisă
<var. num.>	Orice variabilă numerică admisă indiferent de tip (întreg sau real).

3.1. Instrucţiunea REM

Practica programării arată că un program bine comentat este cu mult mai uşor de testat şi pus la zi decît unul necomentat.

În plus, un program bine comentat poate fi uşor înţeles şi utilizat de orice alt programator decît autorul.

De remarcat că textul comentariului poate conţine orice caractere, acesta fiind în întregime ignorat de tratare.

De reţinut că un comentariu pe linie poate fi introdus numai după ultima instrucţiune a liniei ; în caz contrar, textul instrucţiunii este considerat ca făcînd parte din textul comentariului.

Instrucţiunea REM are formatul general :

REM [ARK] <mesaj>

unde prin mesaj se înţelege orice şir de caractere care va fi ignorat la tratare.

Exemplu :

```
10 REM LINIE DE COMENTARIU
```

Pe o linie multiplă, instrucţiunea REM trebuie să fie ultima instrucţiune a liniei.

Exemplu :

```
10 L=2*PI*R : REM LUNGIME CERC
10 L=2*PI*R : 'LUNGIME CERC
```

Linia :

```
100 REM LUNGIME CERC : L=2*PI*R
```

va fi tratată în întregime ca o linie comentariu.

3.2. Instrucţiunea LET

Instrucţiunea LET permite atribuirea la o variabilă simplă sau indexată a unei valori numerice sau şir.

Formatul general al instrucţiunii este :

[LET] <var>=<expr>

Execuţia acestei instrucţiuni constă în evaluarea expresiei din dreapta semnului "=" şi atribuirea valorii astfel determinate variabilei marcate în stînga semnului "=".

Instrucţiunea LET poate fi plasată oriunde în cadrul unei linii multiple.

Exemplu :

```
100 A=SQR (B^2+C^2) : A1=SQR (B^2-C^2)
```

3.3. Instrucţiunea DIM

Limabjul BASIC-PRAE permite utilizarea variabilelor indexate punînd la dispoziţia utilizatorilor mijloace de prelucrare a listelor şi matricilor.

În BASIC-PRAE se pot trata variabile indexate cu unul, doi sau pînă la 255 indici.

Într-un program declararea masivelor se poate realiza pe două căi :

— declararea explicită prin instrucţiunea DIM

— declararea implicită prin prima referinţă la un element de masiv.

La declararea implicită numărul de dimensiuni a masivului declarat este egal cu numărul indicilor variabilei indexate, fiecare dimensiune fiind egală cu 10. Dacă masivul A este declarat implicit printr-o referinţă de forma :

A(<expr.num.>)

atunci va avea o singură dimensiune egală cu 10. Dacă declaraţia implicită se realizează prin :

A(<expr.num.1>, <expr.num.2>)

atunci masivul A va fi un masiv bidimensional avînd dimensiunile (10,10).

Instrucţiunea DIM serveşte la declararea explicită a dimensiunilor unui masiv.

Formatul general :

DIM <var> (<dim1>, <dim2>), <var> (<dim1>, <dim2>)

unde :

var.	reprezintă numele masivului şi poate ii orice nume de variabilă indexată admis în limbaj.
dim1 şi dim2	reprezintă dimensiunile masivului şi pot fi orice expresii numerice admise de limbaj. Numărul dimensiunilor este limitat la 255.

Exemplu :

```
10 N=20 ; M=15
20 DIM A(25), B(N,N). C (10,M)
```

Prin execuţia liniei 10 se declară masivul A avînd dimensiunea (25), masivul bidimensional real B avînd dimensiunile (20, 20) şi masivul bidimensional şir C avînd dimensiunile (10, 15).

Posibilitatea declarării unui masiv cu dimensiuni variabile măreşte mult gama utilizări acestora permiţînd ca la execuţii diferite a unui program spaţiul alocat masivului să aibă lungimi diferite.

Fie programul :

```
10 INPUT N
20 DIM A (N, N), B(N)
100 END
```

Se observă că dimensiunile masivelor A şi B şi deci spaţiul alocat este diferit la execuţii diferite fiind indicate de utilizator prin execuţia instrucţiunii din linia 10.

Declararea explicită a masivelor efectuîndu-se la execuţia instrucţiunii DIM rezultă că aceasta trebuie plasată logic în program înainte de prima referinţă la masivele conţinute, adică înaintea alocării masivului ; în caz contrar, declararea explicită este tratată ca o redimensionare.

De exemplu, fie liniile program :

```
10 A(I+1)=I+1
20 DIM A(100)
```

La execuţia liniei 10 masivul A cu dimensiunea implicită (10) este alocat ; execuţia liniei 20 produce abandonarea execuţiei programului cu indicarea mesajului de redimensionare incorectă.

Se impune deci :

```
10 DIM A(100)
20 A(I+1)=I+1
```

prin care A este declarat cu dimensiunea 100.

Rezultă că este indicată plasarea instrucţiunilor DIM la începutul programelor (ordine logică).

3.4. Instrucţiunea DATA

Instrucţiunea DATA utilizată împreună cu instrucţiunea READ serveşte la introducerea datelor în timpul execuţiei programului.

De remarcat că prin acest cuplu introducerea datelor se face fără intervenţia utilizatorului, fiind dispuse în memorie în blocuri de date odată cu introducerea programului.

Formatul general al instrucţiunii:

DATA <const.>, <const.> ...

Exemplu :

DATA BASIC,'BASIC'

Instrucţiunile DATA pot fi introduse în orice loc într-un program cu condiţia respectării ordinii crescătoare a numărului de linie în funcţie de care se construieşte şi se exploatează blocul de date al programului.

Practica recomandă gruparea instrucţiunilor DATA la sfîrşitul programului pentru verificarea rapidă la testarea şi execuţia programului.

La comanda RUN se iniţializează un pointer către prima dată din blocul de date asociat primei instrucţiuni DATA.

La fiecare moment acest pointer va indica data posibilă de citit prin READ.

3.5. Instrucţiunea READ

Formatul general al instrucţiunii:

READ <var.>, <var.> ....

unde:

<var.> poate fi o variabilă simplă sau indexată de tip numeric sau şir.

La execuţia instrucţiunii READ se face o verificare strictă în privinţa corespondenţei de tip dintre variabila var şi data curent disponibilă.

Dacă tipurile coincid, primei variabile din listă i se atribuie data curentă după care pointerul va indica următoarea dată disponibilă din blocul de date. Dacă în blocul de date al unei instrucţiuni nu mai sînt date disponibile, se trece automat la instrucţiunea DATA următoare dacă aceasta există.

Instrucţiunea READ poate fi plasată oriunde în corpul programului.

Datele introduse prin DATA şi neexploatate prin READ sînt ignorate.

3.6. Instrucţiunea RESTORE

Instrucţiunea RESTORE permite refolosirea prin READ a datelor introduse prin DATA. Acest lucru este necesar cînd într-un program este necesară folosirea în mod repetat a aceloraşi date.

Formatul general al instrucţiunii:

RESTORE <nr. linie>

<nr. linie> indică numărul de linie al instrucţiunii DATA pe blocul de date al căreia se doreşte reiniţializarea pointerului.

Dacă nr. linie nu este indicat, reiniţializarea pointerului se face pe blocul de date al instrucţiunii DATA cu cel mai mic număr de linie.

Exemple:

```
50 RESTORE 1000
100 RESTORE
1000 DATA 'A—B', 5,7, .21
```

3.7. Instrucţiunea INPUT sau LISTEN

Instrucţiunea INPUT oferă utilizatorului posibilitatea introducerii dinamice a datelor necesare executării unui program. Utilizarea datelor prin DATA READ RESTORE este statică deoarece în momentul executării programului acestea sînt îngheţate putînd fi modificate numai prin corectarea unei instrucţiuni DATA.

Folosind instrucţiunea INPUT utilizatorul poate introduce datele necesare execuţiei programului la cererea expresă a acestuia direct pe terminalul de conversaţie. În loc de cuvîntul cheie INPUT se poate folosi şi LISTEN.

Formatul general al instrucţiunii INPUT:

INPUT <const şir>, <var.>, <var.>

unde:

<const.şir> reprezintă orice conatantă şir admisă în limbaj

<var.> reprezintă orice variabilă simplă sau indexată, numerică sau şir, admisă în limbaj

Exemplu:

```
10 INPUT "RAZA CERCULUI:", R
```

Utilizarea <const.şir> în instrucţiunea INPUT măreşte siguranţa introducerii corecte a datelor necesare în fiecare moment. La execuţia liniei program 10 se editează pe linie nouă mesajul :

RAZA CERCULUI :

după care se lansează efectiv cererea de introducere a valorii necesare care se va atribui variabilei R.

Introducerea datelor se consideră terminată în momentul tastării caracterului retur de car.

Dacă numărul datelor introduse este mai mic decît numărul variabilelor din lista instrucţiunii INPUT se relansează pe linie nouă cererea de introducere a datelor suplimentare necesare prin imprimarea caracterului '?'.

Dacă numărul datelor introduse este mai mare decît numărul variabilelor din lista instrucţiunii INPUT datele introduse în exces sînt neglijate cu afişarea mesajului :

*EXTRA LOST

Remarcă : la utilizarea instrucţiunii INPUT datele vor fi separate între ele la introducerea prin caracterul virgulă.

3.8. Instrucţiunea LINE INPUT

Prin convenţia fixată la instrucţiunea INPUT caracterul ',' (virgulă) dispus pe o linie de date cerută de execuţia unei instrucţiuni INPUT are semnificaţia de separator între date. Dacă la execuţia liniei program :

10 INPUT A$

se introduce următoarea linie de date :

INTERPRETOR BASIC, SISTEM DE CALCUL

Variabilei A$ i se atribuie ca valoare constantă şir 'INTERPRETOR BASIC' restul caracterelor de pe linia de date fiind neglijate.

Instrucţiunea LINE INPUT permite atribuirea întregului text al liniei de date introduse (cuprins între ? şi caracterul retur de car) ca valoare a unei variabile şir. Dacă se înlocuieşte linia 10 anterior descrisă cu linia :

10 LINE INPUT A$

prin introducerea liniei de date :

INTERPRETOR BASIC, SISTEM DE CALCUL

variabilei A$ i se atribuie ca valoare constanta şir

'INTERPRETOR BASIC, SISTEM DE CALCUL'.

În contextul instrucţiunii LINE INPUT caracterul ',' (virgulă) îşi pierde semnificaţia de separator, întreaga linie de date (textul cuprins între ? şi retur de car) fiind privită ca constantă şir.

Formatul general al instrucţiunii :

LINE INPUT [<mesaj>], <var.şir>

unde :

<mesaj> reprezintă orice constantă şir încadrată între ghilimele (") ; şi reprezintă mesajul de cerere a liniei.

<var.şir> reprezintă o variabilă simplă sau indexată de tip şir de caractere admisă de limbaj.

La execuţia instrucţiunii LINE INPUT linia de date introdusă are promterul :

? dacă mesaj lipseşte altfel apare promterul <mesaj> ?.

3.9. Instrucţiunea PRINT

Instrucţiunea PRINT este utilizată pentru editarea anumitor date rezultate intermediare sau finale ale unei execuţii la terminalul utilizatorului. Formatul general al instrucţiunilor este :

PRINT <lista>

sau

PRINT AT <expr1>, <expr2> ; <lista>

unde :

<lista> reprezintă o listă de elemente admise de limbaj separate între ele prin caracterele ',' (virgula) sau ';' (punct virgulă).

<expr1>,<expr2> reprezintă coordonatele x,y a ecranului începînd de unde se doreşte scrierea. În cazul cînd ieşirea nu este pe ecran această parte a instrucţiunii este inefectivă. Valoarea x se trunchează la multiplu de 8.

Elementele unei liste PRINT pot fi :

— expresii numerice sau şir care se pot reduce la constante sau variabile ;
— TAB (expr. num.) unde expr. num. poate fi orice expresie numerică admisă de limbaj ;
— SPC (expr. num.)

O succesiune de două caractere separator semnifică absenţa elementului de tipărit. Lista de elemente se poate termina prin separator.

Prin execuţia operatorului PRINT se construieşte un text care se transmite perifericului. Acest text este rezultatul prelucrării succesive a elementelor de tipărit şi a separatorilor.

Prin editarea expresiilor şir se construieşte o secvenţă constituită din toate caracterele valorii expresiei şir.

Exemplu :

```
10 PRINT "BASIC-PRAE" + "CLUJ-NAPOCA"
```

va extrage pe terminal

```
BASIC-PRAE CLUJ-NAPOCA
```

Prin editarea unei expresii aritmetice se construieşte o succesiune de simboluri care constituie reprezentarea zecimală a valorii expresiei. Convenţiile de editare în formatul implicit sînt :

— reprezentarea zecimală a unui număr este precedată de spaţiu (nr. pozitiv) sau semnul minus (număr negativ), şi urmată de blanc.

— precizia reprezentării numerelor este de 11 cifre semnificative.

Orice număr care poate fi reprezentat ca întreg va fi extras ca număr zecimal întreg (fără punct).

Numerele a căror valoare absolută este cuprinsă între 1/100 şi 1 000 000 şi se pot reprezenta exact sub forma unui număr zecimal, se reprezintă cu format zecimal. Zerourile nesemnificative nu se reprezintă. Restul numerelor se reprezintă în format zecimal cu exponent sub forma:

semn mantisa E semn exponent spaţiu

cu mantisa cuprinsă între 1 şi 10 avînd cel mult 11 cifre semnificative, iar exponentul cuprins între 0 şi 38.

De exemplu :

— valoarea 625 se va extrage sub forma : 625
— valoarea 0.012345 se va extrage sub forma .012345
— valoarea 0.00123456 se va extrage sub forma :

1.23456E-03

Liniile care vor fi extrase se împart în zone a 14 caractere (5 cîmpuri pentru un terminal cu 80 caractere, respectiv 8 cîmpuri pentru linia cu 132 caractere).

Separatorul ';' produce generarea în textul de tipărit a unui şir vid. Textele generate din două valori de elemente separate prin ';' se succed nemijlocit.

Separatorul ',' produce generarea în textul de ieşire după valoarea elementului care-l precede a unuia sau a mai multor caractere blanc (spaţiu) pînă la umplerea zonei respective.

Existența unui separator (',' sau ';') după ultimul element al listei instrucțiunii PRINT suprimă operația RETUR DE CAR și salt la linie nouă. În acest fel instrucțiunea PRINT imediat următoare va edita datele pe aceeași linie terminal.

Exemplu:

```
10 R=2. : PRINT "RAZA=";R;
15 PI=4*ATN(1)
20 PRINT "ARIA CERCULUI="; PI*R**2
RUN
RAZA=2 ARIA CERCULUI=12.566370614
```

Instrucțiunea PRINT fără listă extrage pe terminal tot șirul creat anterior, iar dacă nu există scrie o linie vidă.

Pentru un control mai eficient al formatului de editare, limbajul BASIC PRAE pune la dispoziția utilizatorilor funcția de tabulare TAB.

Funcția TAB acceptă ca parametru orice expresie numerică. La execuție se evaluează valoarea expresiei și dacă este necesar se realizează conversia real-întreg pentru a obține o valoare întreagă care va reprezenta numărul coloanei curente în care se va scrie în continuare. Deoarece numărul coloanei curente are o evoluție strict progresivă rezultă că funcția TAB este inefectivă pentru argumente negative, nule sau mai mici decît numărul coloanei curente. Dacă valoarea transmisă ca parametru este mai mare decît lungimea unei linii terminal, funcția TAB provoacă extragerea liniei curente și poziționarea la începutul liniei terminal imediat următoare. Instrucțiunea PRINT poate provoca extragerea uneia sau mai multor linii terminal.

Funcția SPC acceptă ca parametru orice expresie numerică. Funcția generează atîtea spații cît este valoarea expresiei trunchiată la întreg.

Instrucțiunea PRINT poate fi folosită în mod imediat și poate să apară în orice poziție pe o linie multiplă.

3.10. Instrucțiunea PRINT USING

Dacă într-un program se dorește editarea datelor altfel decît cu formatul implicit, limbajul BASIC-PRAE pune la dispoziția utilizatorului instrucțiunea PRINT USING.

Formatul general al instrucțiunii este:

PRINT USING <format>, <listă>

PRINT AT <e1>, <e2> : USING <format>, <listă>

unde:

<format>	reprezintă o expresie șir ce cuprinde formatul de editare
<listă>	reprezintă o listă de elemente separate între ele prin virgulă sau punct-virgulă. Din punct de vedere sintactic lista este identică cu lista de la instrucțiunea PRINT.
<e1>, <e2>	sînt coordonatele x,y ale ecranului începînd de unde se dorește tipărirea.

Instrucțiunea PRINT USING asigură editarea datelor, rezultatelor la terminalul utilizatorului sub forma specificată la format.

În timpul execuției instrucțiunii, lista este parcursă în paralel cu șirul format. La terminarea tratării listei caracterele rămase netratate în șir dacă există, vor fi extrase așa cum apar. La terminarea șirului format elementele rămase în listă sînt editate cu formatul dat.

Exemplu:

```
PRINT USING "##·##ESTE SOLUŢIA", 12·34 ; 12·34E-2
12·34·12 ESTE SOLUŢIA
```

Separatorii dintre elementele listei nu își păstrează semnificația la instrucțiunea PRINT; funcția TAB nu mai poate figura în listă.

Șirul format este interpretat caracter cu caracter și poate conține orice caracter admis de limbaj. Pentru definirea formatului anumite caractere au semnificație specială (prin convenție). Caracterele cu semnificație specială sînt: (diez), (virgulă), * (asterisc), · (punct),

$ (dolar), ^ (circumflex), — (minus), ' (apostrof), şi în anumite forme literele : L, R, C, E. Orice alt caracter diferit de cele înşirate va fi editat aşa cum apare în format :

Exemplu :

```
PRINT USING "ALFABETA/DELTA",X
ALFABETA/DELTA O
```

3.10.1. *Editarea elementelor numerice*

PRINT USING editează elementele numerice conform formatului şir specificat. Convenţiile pentru format sînt :

— caracterul # (diez) plasat în format specifică un cîmp numeric. Numărul caracterelor # apărute consecutiv specifică numărul cifrelor ocupate de număr cadrat la dreapta. Cîmpurile neocupate se umplu cu blancuri.

Exemplu :

```
PRINT USING"####   #######",1234 ; 1·E6
1234 1000000
PRINT USING"####",12
12
```

Dacă numărul nu încape în cîmpul specificat numărul va fi extras cu format implicit

Exemplu :

```
PRINT USING"##",135
%135
```

Numerele vor fi rotunjite dacă este cazul.

Exemplu :

```
PRINT USING"## ##",12·4 ; 12·5
12 13
```

Dacă se doreşte editarea unui număr cu punct zecimal, se introduce caracterul (punct) în şirul format. Numărul caracterelor dispuse la stînga punctului zecimal în şirul format indică numărul de cifre zecimale a părţii întregi a numărului ; cele dispuse la dreapta punctului zecimal dau numărul de cifre a părţii zecimale a numărului.

Exemplu :

```
PRINT USING"#·# ·# #·",1·1 ; ·1 ; 1·6
1·1 ·1 2·
```

Pentru editarea numerelor negative trebuie să fie introdus în şirul format un caracter pentru semn :

Exemplu :

```
PRINT USING"##·## ##·##",—1 ; —2
—1·00 —2·00
```

Dacă se doreşte editarea semnului la dreapta numărului, atunci şirul caracterelor se termină cu caracterul — (minus) :

Exemplu :

```
PRINT USING"##·#— ##·##—",1·0 ; —1·0
1·0 1·0—
```

Dacă în şirul format două caractere * (asterisc) preced formatul numeric, atunci cîmpul este completat la stînga de asteriscuri :

Exemplu :

```
PRINT USING "SOLUŢIA :X=**##",1
SOLUŢIA X=***1
```

Numerele negative pot fi extrase cu acest format. ,sau se folosește — la sfîrșit de format ;

Exemplu :

```
PRINT USING "**##",-1
*-1
PRINT USING "**##-",-1
**1-
```

Dacă formatul numeric este precedat de două caractere $ (dolar), la editare numărul va fi precedat de semnul $ (dolar). Numerele negative pot fi extrase cu acest format :

Exemplu :

```
PRINT USING "PRET=$$##",12
PRET=$12
PRINT USING "PRET=$$##", —1
PRET=—$1
PRINT USING "PRET=$$##—",—1
PRET= $ 1—
```

Plasarea caracterului , (virgulă) oriunde în stînga punctului zecimal între caracterele în cadrul formatului produce separarea prin virgulă în grupe de 3 (trei) cifre din stînga punctului zecimal a numărului care trebuie editat.

Exemplu :

```
PRINT USING"##, ########$$#, ####",1.E6,12345—6
1,000,000 $ 12,346
```

Formatul exponențial este specificat prin caractere (circumflex) după formatul numeric. Formatul exponențial nu admite caracterele *, $ și —. Dacă șirul format conține mai puține caractere decît 4, ele vor apare la editare așa cum sînt specificate la format. În cazul cînd un element din listă nu încape în cîmpul specificat și este editat în mod implicit se abandonează tratarea șirului format în continuare și toate elementele vor fi extrase cu formatul implicit. Primul cîmp caracter din format este rezervat pentru semn.

Exemplu :

```
PRINT USING "##^^^^", 12·56E20
1E+20
PRINT USING
". ^^^^ . ^^^ ^^^^ ", 1E7 ; 13·2 ; 1E8
1·0E+07 13·2 ^^^ 1·0E+08
```

Formatul exponențial nu este compatibil cu descriptorii '$' și '*'.

3.10.2. *Editarea șirurilor prin PRINT USING*

Formatul de editare șir furnizează informații privind : numărul de caractere alfanumerice de extras, cadrarea șirurilor (stînga, dreapta).

Descrierea formatului șir începe totdeauna cu caracterul '(apostrof) și se poate continua cu un șir de caractere identice (caracterul repetat poate fi 'C', 'E', 'L', 'R'). Lungimea cîmpului șir indicat de un format șir este dat de numărul de repetiții din format mărit cu unu.

În cazul în care lungimea șirului de extras este mai mare decît lungimea cîmpului șir descris de format, la editare se vor extrage începînd de la stînga atîtea caractere cîte indică lungimea cîmpului. Dacă lungimea șirului este mai mică decît lungimea șirului format atunci șirul va fi extras în întregime cadrat conform descriptorului și completat cu blancuri. În situația în care formatul conține numai caracterul apostrof ('), se va edita primul caracter din șir :

Exemplu :

```
PRINT USING "'", "ABCD"
A
```

Formatul şir cu indicaţia cadraj la stînga este constituit din caracterul apostrof (') urmat de o serie de caractere "L". În acest caz şirul se editează începînd cu primul caracter din stînga, de lungime egală cu numărul caracterelor "L" plus 1.

Exemplu :

```
PRINT USING " 'LLLL", "ABC"
ABC
PRINT USING " 'LLLL", "LMNOPRS"
LMNOP
```

Formatul şir cu indicaţia cadraj la dreapta este format din caracterul apostrof ('), urmat de o serie de caractere "R". În acest caz şirul se editează cadrat la dreapta completat eventual la stînga cu blancuri.

Dacă lungimea şirului depăşeşte lungimea cîmpului şirul va fi extras cadrat la stînga :

Exemplu :

```
PRINT USING " 'RRRR","ABC"
 ABC
PRINT USING " 'RRRR","ABCDE"
ABCDE
PRINT USING " 'RRRR","BASIC-PROE CLUJ-NAPOCA"
BASIC
```

Formatul centrat constă dintr-un caracter apostrof (') urmat de o serie de caractere "C". Cu acest format se editează un şir pe cîmpul format cadrat central şi completat cu blancuri în stînga şi dreapta.

Dacă lungimea şirului este mai mare decît lungimea cîmpului format şirul va fi cadrat la stînga.

Exemplu :

```
PRINT USING " 'CCCCCCC", "ABC"
  ABC
PRINT USING " 'CCCCCCC","ABCD"
 ABCD
PRINT USING " 'CCCCCCC","ABCDEFGHIJKLMNO"
ABCDEFGH
```

Formatul extins constă dintr-un caracter apostrof (') urmat de o serie de caractere "E". Cu acest format se editează un şir pe cîmpul format, cadrat la stînga.

Dacă lungimea şirului depăşeşte cîmpul specificat cîmpul va fi extins şi tot şirul va fi extras :

Exemplu :

```
PRINT USING " 'EEE****","A"
A ****
PRINT USING ",EEE****","ABCD"
ABCD****
PRINT USING " 'EEE****","ABCDEFGH"
ABCDEFGH****
```

3.11. Instrucţiunea GOTO

Instrucţiunea GOTO serveşte la modificarea ordinii secvenţiale de execuţie a instrucţiunilor program, (salt necondiţionat).

Formatul general al instrucţiunii :

GOTO <nr. linie>

unde nr. linie este orice număr de linie admis (23767).

La execuţia acestei instrucţiuni controlul este transferat primei instrucţiuni a liniei indicate. Din formatul specificat este evident că nu se poate da controlul altei instrucţiuni decît la prima de pe o linie program :

Exemplu :

```
50 GOTO 200
200 X=X+1 : PRINT X
```

În exemplul dat controlul nu poate fi transferat instrucţiunii PRINT.

Dacă prima instrucţiune a liniei specificate este neexecutabilă (DATA, REM) controlul va fi transferat primei instrucţiuni care urmează.

Exemplu :

```
10 GOTO 50
50 DATA 1,2,3
60 X=A 2+B 2
```

În acest caz, controlul va fi transferat instrucţiunii X=A2+B2. Încercarea de transfer a controlului la o linie inexistentă se soldează cu oprirea execuţiei şi afişarea unei erori.

Instrucţiunea GOTO poate avea orice poziţie pe o linie multiplă.

3.12. Instrucţiunea ON . . . GOTO

Instrucţiunea ON. . . GOTO serveşte la realizarea transferului selectiv conform valorii unei expresii. Formatul general al instrucţiunii este :

ON <expr. num.> GO TO <nr. linie>, <nr. linie>, ...

La execuţia instrucţiunii se evaluează expresia numerică iar partea întreagă a valorii calculate se foloseşte ca index în lista numerelor de linie pentru a determina linia căreia i se predă transferul. La execuţia instrucţiunii :

ON X GOTO 100,200,300

controlul este transferat liniei 100 dacă X are valoarea 1, liniei 200 dacă valoarea liniei X este 2 liniei 300 dacă X are valoarea 3. În cazul în care partea întreagă este negativă, nulă, sai mai mare decît numărul elementelor din listă, controlul trece la instrucţiunea ce urmează în secvenţă.

Instrucţiunea ON . . . GOTO poate avea orice poziţie pe linia multiplă.

3.13. Instrucţiunea IF

Instrucţiunea IF serveşte pentru realizarea unui transfer al controlului sau execuţia unei instrucţiuni în funcţie de valoarea logică a unei condiţii. Formatul general al instrucţiunii :

IF <cond.> THEN <nr. linie>, <instr.> [ELSE <nr.linie>, <instr.>]

unde :

<cond.>	poate fi orice expresie logică de relaţie sau aritmetică.
<nr. linie>	poate fi orice număr de linie admis
<instr.>	poate fi orice instrucţiune alta decît :
	FOR, NEXT, FNEND, END, DIM, DEF, IF, REM, DATA.

Dacă valoarea logică a condiţiei este adevărată atunci se execută partea THEN (sau GOTO) a instrucţiunii, altfel se execută partea ELSE a instrucţiunii :

IF A>B THEN PRINT "ADEVĂRAT" ELSE PRINT "FALS"

3.14. Cicluri

Prin ciclu se înțelege un set de instrucțiuni a căror execuție se repetă pînă la îndeplinirea unei condiții. Fiecare ciclu posedă elemente caracteristice :

— variabilă de ciclu ;
— valoarea inițială ;
— test final.

Exemplu :

```
100 I=1
110 IF I 10 THEN 140
120 PRINT I, I 2, I 3
130 I=I+1 : GO TO 110
140 PRINT "SFIRSIT CICLU"
```

Fiecare ciclu are patru părți :

— inițializarea ciclului prin care se inițializează variabila ciclului. (linia 100) ;
— condiția de terminare a execuției ciclului (linia 110) ;
— corpul ciclului (linia 120).
— modificarea valorii variabilei cu pasul ciclului.

Limbajul BASIC-PRAE prin intermediul instrucțiunilor FOR și NEXT permite declararea și constituirea comodă și simplă a cicurilor.

3.15. Instrucțiunea FOR · · · TO

Instrucțiunea FOR definește începutul ciclului, formatul general al instrucțiunii FOR :

FOR <var.num> = <expr.num1> TO <expr.num2> STEP <expr.num3>

unde :

<var. num.>	poate fi orice variabilă simplă și reprezintă variabila de ciclu.
<expr. num1>	poate fi orice expresie numerică, valoarea ei reprezentînd valoarea inițială a variabilei ciclului.
<expr. num2>	poate fi orice expresie numerică, valoarea ei reprezentînd valoarea finală a variabilei ciclului.
<expr. num3>	poate fi orice expresie numerică, valoarea ei reprezentînd valoarea pasul cu care se modifică variabila ciclului.

În cazul în care indicarea pasului lipsește, pasul de ciclare se consideră implicit 1.
Variabila de control a ciclului poate fi modificată în interiorul ciclului.
Expresiile care apar pe instrucțiunea FOR sînt evaluate o singură dată la execuția instrucțiunii FOR.

3.16. Instrucțiunea NEXT

Instrucțiunea NEXT definește sfîrșitul ciclului. Formatul general al instrucțiunii este :

NEXT <var.num.>

unde <var.num.> este variabila cu același nume ca și pe instrucțiunea FOR corespunzătoare. La execuția instrucțiunii NEXT se incrementează valoarea variabilei ciclului cu valoarea pasului și se face testul final al ciclului. Ciclurile FOR ... NEXT pot fi îmbricate adică corpul unui ciclu poate să conțină alt ciclu.

3.17. Subrutine

Limbajul BASIC-PRAE permite declararea unui set de instrucțiuni ca o subrutină prin intermediul a două instrucțiuni :

— Instrucțiunea GOSUB de apel a subrutinei.

— instrucţiunea RETURN, de retur din subrutina la instrucţiunea imediat următoare apelului.

Într-un program pot fi definite mai multe subrutine. Acestea pot fi plasate oriunde în corpul programului.

3.17.1. *Instrucţiunea GOSUB, instrucţiunea RETURN*

Instrucţiunea GOSUB provoacă lansarea execuţiei unei secvenţe de instrucţiuni declarate ca subrutină a programului, cu reţinerea instrucţiunii care urmează în secvenţa instrucţiuni de apel.

Formatul general :

GOSUB <nr. linie>

În BASIC-PRAE nu există posibilitatea declarării explicite a unei subrutine, mai precis a începutului subrutinei. Controlul poate fi transferat oricărei instrucţiuni. Odată controlul transferat unei subrutine, acesta se execută pînă la întîlnirea instrucţiunii RETURN. Instrucţiunea RETURN asigură marcarea sfîrşitului subrutinei şi revenirea în program la instrucţiunea care urmează după instrucţiunea GOSUB de apel.

Formatul general :

RETURN

Transferul controlului dintr-o subrutină spre exteriorul acesteia este permis, dar este mai indicată ieşirea prin instrucţiunea RETURN.

Ca şi ciclurile program, subrutinele pot fi imbricate :

Exemplu :

```
10 REM IMBRICARI SUBRUTINE
20 GOSUB 100
30 STOP
100 PRINT "SINT IN SUBRUTINA 1"
110 GOSUB 200
120 RETURN
200 PRINT "SINT IN SUBRUTINA 2"
210 RETURN
```

O subrutină poate apela orice subrutină chiar şi pe ea însăşi. Profunzimea imbricării depinde numai de spaţiul de memorie pus la dispoziţia utilizatorului.

O subrutină poate conţine mai multe instrucţiuni de revenire (RETURN).

3.17.2. *Instrucţiunea ON ··· GOSUB*

Instrucţiunea ON ... GOSUB permite transferul calculat al controlului la una din subrutinele indicate.

Formatul general al instrucţiunii :

ON <expr. num.> GOSUB <nr. linie>, <nr. linie> ....

La execuţia instrucţiunii se evaluează expresia numerică, partea întreagă a valorii calculate este folosită ca index în lista numerelor de linie. Controlul va fi transferat subrutinei indicată de index. Execuţia instrucţiunii RETURN provoacă returul în program la instrucţiunea care urmează după ON ... GOSUB.

Instrucţiunea ON ... GOSUB poate servi la apelarea unei subrutine prin unul din punctele sale de intrare.

Exemplu :

```
50 ON X GOSUB 100,200,300
100 REM INCEPUTUL SUBRUTINEI
200 REM AL DOILEA PUNCT DE INTRARE
300 REM AL TREILEA PUNCT DE INTRARE
500 RETURN
```

Controlul este transferat liniei 100 dacă X este egal cu 1, liniei 200 dacă X are valoarea 2, liniei 300 dacă X=3, și instrucțiunii care urmează după ON ... GOSUB dacă X=0 sau X=4.

3.18. Funcții definite utilizator

Practica programării arată că deseori într-un program apare necesitatea repetării, în diverse puncte, a unei expresii matematice sau șir, sau a unui set de intrucțiuni, ceea ce duce la mărirea spațiului de memorie ocupat de program.

Pentru rezolvarea acestei dificultăți BASIC-PRAE pune la dispoziție posibilitatea definirii funcțiilor care se pot apela ca și funcțiile standard.

Funcțiile utilizator se definesc prin instrucțiunea, DEFFN. Funcția utilizator poate fi :

— funcție simplă definită a cărei definiție este data printr-o instrucțiune DEF FN.

— funcție multilinie a cărei definiție este reprezentată de un set de instrucțiuni cuprins între instrucțiunile DEFFN și FNEND.

3.18.1. *Funcția simplu-definită*

Formatul general al instrucțiunii DEF FN pentru definirea unei funcții simplu definite este :

```
DEF FN <fct> (<par. form.>, ...) = <expr.>
```

unde :

<fct> poate fi orice nume de funcție admis inclusiv tipul ($).

<par. formal> poate fi orice nume de variabilă simplă sau șir.

<expr.> poate fi orice expresie de același tip cu tipul funcției.

Exemplu :

```
DEF FNA (X) = X 2+X-1
```

Expresia unei funcții poate conține un apel la orice funcție standard sau utilizator definită în prealabil :

Exemplu :

```
DEF FN B (X, Y)=Y 2-INT (FNA (X))
```

De reținut că tipul expresiei de definiție (numerică sau șir) trebuie să coincidă cu tipul declarat al funcției (numerică sau șir). De exemplu instrucțiunea :

```
DEF FNC $ (X $, Y)=LEN (X $ )+Y
```

nu este admisă.

Orice variabilă care apare în expresia funcției și nu apare în lista variabilelor formale va fi considerată ca fiind o variabilă a programului în care este definită funcția. La apelul unei funcții definite se face o verificare strictă în privința coincidenței între numărul și tipul parametrilor actuali și numărul și tipul parametrilor formali.

3.18.2. *Funcția multilinie*

Formatul definiției unei funcții multilinie este următorul :

```
DEFFN <fct.> (<par. form> ...)
Instrucțiuni care definesc corpul definiției
FNEND <val. funcției>
```

Instrucțiunea DEFFN pentru funcția multilinie se deosebește de cea pentru funcția simplă defnită prin absența semnului '='.

Valoarea produsă la apelul unei funcții multilinie este valoarea expresiei de pe instrucțiunea FNEND.

În corpul definiţiei unei funcţii nu au sens instrucţiunile : DATA, DEF, DIM, END.

Definiţia unei funcţii multilinie poate să apară oriunde în program, cu condiţia ca execuţia ei trebuie să preceadă primul apel al funcţiei.

Din corpul unei funcţii multilinie se poate ieşi şi cu instrucţiunea FNRETURN, care are sintaxa :

FNRETURN <expresie rezultat>

Variabilele ce apar în corpul definiţiei şi nu sînt parametri formali ai funcţiei sînt considerate ca fiind variabile ale programului apelant.

Exemplu : (definiţia funcţiei factorial)

```
10 DEF FNF (N)
20 IF N=1 THEN FNRETURN 1
30 FNEND FNF (N—1)*N
40 INPUT N : PRINT N;" ":"="; FNF(N)
```

Funcţia multilinie poate fi de tip numeric sau de tip şir.

Parametrii formali pot avea orice tip. La apelul unei funcţii definite se face o verificare strictă privind numărul şi tipul parametrilor de apel (actuali) şi numărul şi tipul parametrilor formali.

Instrucţiunile din corpul definiţiei unei funcţii multilinie pot fi introduse în orice ordine dar la introducere trebuiesc încadrate între DEFFN şi FNEND care stabilesc cîmpul de acţiune al parametrilor formali.

3.19. Instrucţiunea RANDOMIZE

Formatul instrucţiunii :

RANDOMIZE

Instrucţiunea RANDOMIZE poate fi plasată oriunde în program dar ea trebuie să fie totuşi la începutul programului, practica recomandă utilizarea instrucţiunii RANDOMIZE după punerea la punct a programului.

3.20. Instrucţiunile STOP şi END

Instrucţiunile STOP şi END sînt utilizate la oprirea execuţiei programului. Formatul general al instrucţiunii END :

END

Instrucţiunea END poate fi dispusă şi pe linie multiplă.

Instrucţiunea STOP suspendă temporar execuţia unui program care poate fi oricînd reluată începînd cu instrucţiunea următoare printr-o comandă CONTINUE sau începînd cu o instrucţiune oarecare specificată prin GOTO <nr. linie> în mod imediat.

Formatul general :

STOP

Instrucţiunea STOP poate să apară în orice punct al programului şi pe orice poziţie a unei instrucţiuni multiple. La execuţia instrucţiunii programul este oprit cu mesajul :

[_*BREAK LINE N

N fiind numărul de linie pe care se află STOP. Execuţia programului putînd fi reluată în orice moment.

Instrucţiunea STOP oferă un mijloc foarte eficient de punere la punct a programelor cînd este utilizată în combinaţie cu modul imediat de lucru prin care se pot testa elementele programului.

3.21. Funcţia INP

BASIC-PRAE pune la dispoziţia utilizatorului funcţia :

INP (<arg>)

pentru citirea unui port. Argumentul funcţiei este numărul portului iar caracterul citit este atribuit valorii A în exemplu :

A=INP (O)

3.22. Instrucţiunea OUT

BASIC-PRAE pune la dispoziţia utilizatorului instrucţiunea OUT pentru a scrie un caracter pe un port. Formatul general al instrucţiunii este :

OUT <nr. canal>, <expr>.

Instrucţiunea dă valoarea <expr.> canalului specificat de <nr. canal>.

3.23. Funcţia PEEK

BASIC-PRAE permite utilizatorului accesul la memoria fizică. Sintaxa funcţiei este :

PEEK (<adresa>)

unde :

<adresa> este o valoarea care reprezintă adresa zecimala a locaţiei de memorie care urmează să fie citită.

Exemplu :

B=PEEK (A)

conţinutul adresei A este atribuit variabilei B.

3.24. Instrucţiunea POKE

BASIC-PRAE permite utilizatorului scrierea în memoria fizică. Sintaxa instrucţiunii este :

POKE <adresa>, <valoare>

Prin instrucţiunea POKE valoarea specificată de octetul <valoare> este atribuită locaţiei de adresă dată de <adresă>.

3.25. Instrucţiunea BEEP

Instrucţiunea BEEP serveşte la producerea unui sunet cu ajutorul generatorului sonor. Sintaxa instrucţiunii este :

BEEP <durata>, <frecvenţa>

unde <durata> este o valoare cuprinsă între 0 şi 255 şi stabileşte durata sunetului ; <frecvenţa> este o valoare cuprinsă între 0 şi 32767 şi stabileşte frecvenţa sunetului.

3.26. **Instrucţiunea CALL**

BASIC-PRAE permite utilizatorului apelarea unor subprograme scrise în limbaj de asamblare (cod maşină).

Formatul instrucţiunii este :

CALL <adresă>, <arg. 1>, <arg. 2>, ...

Instrucţiunea predă controlul adresei specificate. Fiecare argument se reprezintă pe doi octeţi şi schema de transmitere a lor este următoarea :

Regiştrii : HL adresa primului agrument de pe STACK.
BC conţine numărul argumentelor aflate pe STACK
SP conţine adresa vîrfului STACK-ului unde se află argumentele astfel :
ARG.N ← SP
...
ARG.2
ARG.1 ← HL
ADRESA DE RETUR

3.27. **Funcţia FRE**

Funcţia FRE are rolul de a furniza utilizatorului memoria liberă. Dacă parametrul este o variabilă, se dă memoria disponibilă pentru variabile iar dacă se dă o variabilă şir se furnizează zona liberă şir .

Exemplu :

FRE(X) furnizează zona liberă variabile.
FRE(X$) furnizează zona liberă şir.

4. MODUL IMEDIAT

Pentru rezolvarea unor probleme simple reduse la calcularea unor expresii folosind BASIC-PRAE nu se impune scrierea unui program complet şi apoi lansarea execuţiei acestuia.

Majoritarea instrucţiunilor limbajului BASIC-PRAE pot fi utilizate ca nişte comenzi "ON LINE" şi executate imediat după introducere.

Apar astfel două moduri de lucru care se pot folosi separat sau combinate dacă se utilizează sub sistemul BASIC-PRAE :

— modul imediat (direct) ;
— modul program (indirect).

Instrucţiunea folosită în mod imediat, executîndu-se după introducere, rezultă că nu poate fi folosită pe o linie multiplă ; de aici apare echivalenţa dintre instrucţiunea imediată şi linia imediată.

Deosebirea dintre linia program şi linia imediată constă în faptul că linia imediată nu posedă număr de linie.

Exemplu : linia :

10 PRINT "BASIC-PRAE"

este o linie program, în timp ce linia :

PRINT "BASIC-PRAE"

este o linie imediată.

Liniile care încep cu un număr de linie sînt memorate într-o forma internă specială şi executate ulterior la o comandă RUN.

Modul de lucru imediat este deosebit de util în două situaţii tipice :

— punerea la punct a programelor;

— efectuarea unor calcule care nu ar justifica scrierea unui program.

Folosirea anumitor instrucţiuni în mod imediat nu are sens. Din această categorie fac parte instrucţiunile DATA, DEF, FNEND.

Posibilitatea marcării într-un program a punctelor de întrerupere, prin utilizarea instrucţiunii STOP, pune la dispoziţia utilizatorilor un mijloc foarte eficace de punere la punct a programelor prin combinarea judicioasă a celor două moduri de lucru. În plus execuţia unui program întrerupt prin STOP poate fi oricînd reluată printr-o comandă CONTINUE sau printr-o instrucţiune GO TO în mod imediat.

5. COMENZI BASIC-PRAE

Introducerea unui program în memorie nu este suficientă pentru rezolvarea problemei pentru care acesta a fost proiectat.

Practica programării arată că sînt foarte rare situaţiile în care un program scris este absolut corect. Apare astfel problema punerii la punct a unui program introdus. Există situaţii în care punerea la punct a unui program necesită listarea acestuia sau ştergerea şi corectarea unor linii. Odată pus la punct pentru o utilizare ulterioară apare problema salvării programului pe un suport de unde să poată fi regăsit în orice moment. Salvările succesive ale programelor pot duce la lipsa spaţiului pe suport extern pentru salvări. Apare astfel problema eliminării unor programe care nu mai sînt utilizate.

Pentru rezolvarea dificultăţilor mai sus enumerate BASIC-PRAE pune la dispoziţia utilizatorilor un set de instrucţiuni speciale numite comenzi. Din punct de vedere al tratării o comandă poate fi privită ca o instrucţiune imediată, ele pot să apară şi în program.

Linia de comandă este identică cu linia imediată fiind lipsită de număr de linie ; deoarece se execută imediat după introducerea ei poate fi şi linie multiplă. Între comenzi şi instrucţiuni există însă o deosebire esenţială constînd în aceea ca în timp ce o instrucţiune operează asupra entităţilor dintr-un program, comanda operează asupra programului privit ca o entitate de sine stătătoare.

În funcţie de tipul de operaţii executate asupra programului, comenzile se împart în trei categorii :

a) comenzi de prelucrare a programului curent în memoria pusă la dispoziţia utilizatorului (comenzi de editare) :

1. DELETE
2. LIST
3. NEW
4. RENUMBER
5. EDIT
6. AUTO

b) comenzi de execuţie şi asistare a execuţiei unui program :

8. CONTINUE
9. TRACE
10. RUN

c) comenzi de salvare (încărcare a programelor în) din bibliotecile de programe :

12. AMERGE
13. LENGTH
14. LOAD, ALOAD
15. PRECISION
16. SAVE, ASAVE
17. KILL

Formatul general al unei linii de comandă este :

<cuvînt cheie> <argumente>

Fiecare dintre paragrafele care urmează conţin informaţii complete privind comanda pe care o tratează.

5.1. Comenzi de editare

În categoria comenzilor de editare intră comenzile prin care se realizează ştergerea unui set de instrucţiuni, listarea parţială sau totală.

Aceste comenzi nu influenţează execuţia unui program ci pregătesc spaţiul de memorie al utilizatorului şi servesc la corectarea programului.

5.1.1. *Comanda DELETE*

Comanda DELETE serveşte la eliminarea dintr-un program a unui set de instrucţiuni continue din punctul de vedere al numărului de linie. În fapt, comanda realizează eliminarea unui set de linii program

Formatul general al comenzii:

DELETE N1—N2

unde N1, N2 poate fi orice număr de linie admis de limbaj. Se impune însă respectarea strictă a condiţiei :

N1=N2

Comanda realizează eliminarea din program a liniilor avînd numărul de linie cuprins în intervalul închis N1, N2.

Dacă N1=N2 comanda realizează eliminarea liniei cu numărul N1 (=N2) dacă aceasta există.

5.1.2. *Comanda LIST*

Comanda LIST serveşte la extragerea pe terminalul utilizatorului a programului sau a unei părţi din programul curent dispus în memoria internă.

Formatul general al comenzii :

LIST {N-/-N/N1-N2}

unde N, N1, N2 poate fi orice număr de linie admis de limbaj.

În formatul :

LIST N-

comanda realizează extragerea la terminalul utilizatorului a tuturor liniilor programului avînd numărul de linie mai mare sau egal cu N-. În formatul :

LIST -N

comanda realizează extragerea la terminalul utilizatorului a tuturor liniilor programului avînd numărul de linie mai mic sau egal cu N. În formatul :

LIST N1-N2

comanda realizează extragerea la terminalul utilizatorului a liniilor programului avînd numărul de linie cuprins în intervalul închis N1, N2.

De reţinut că în acest format se impune condiţia N1<=N2. Dacă N1=N2 se editează linia cu numărul N1(=N2) dacă aceasta există.

Listarea programului la un alt terminal nu poate fi realizată prin comanda LIST ci numai prin comanda LLIST.

Pentru a lista toate variabilele programului cu valorile corespunzătoare, utilizatorul are la dispoziţie comanda LVAP

Sintaxa comenzii este :

LVAR

Dacă utilizatorul dorește să scrie (afișeze) pe imprimantă, atunci BASIC-PRAE îi pune la dispoziție următoarele instrucțiuni și comenzi echivalente cu comenzile din paranteză :

LLIST (LIST) ; LLVAR (LVAR) ; LNULL (NULL) ;
LPRINT (PRINT) ; LPRINT USING (PRINT USING);
LTRACE (TRACE) ; LWIDTH (WIDTH) ; LPOS (POS)

Pentru a fixa lungimea rîndului de editare, BASIC-PRAE pune la dispoziția utilizatorului comanda WIDTH avînd formatul :

WIDTH <arg.>

unde <arg.> este lungimea liniei.

5.1.3. *Comanda TRACE*

Comanda TRACE cuplează modul de trasare a programului pentru a se putea urmări urma programului. Sintaxa comenzii este :

TRACE <expresie>

Dacă valoarea <expresie> -I este diferită de O va fi trasat (urmărit) ; adică se vor afișa numerele de linie ale liniilor executate. Dacă <expresie> are valoarea O programul va fi executat fără facilitatea TRACE.

5.1.4. *Comanda EDIT*

Comanda EDIT servește la corectarea unei linii BASIC. Comanda are formatul general :

EDIT<nr. linie>

Unde:

<nr. linie> este numărul de linie al liniei ce urmează să fie corectată.

La această comandă mai există un set de comenzi speciale ce constă din cifre și litere, care nu sînt afișate la tastare. Numerele pot fi în intervalul 1,255.

Lista comenzilor este următoarea :

A	încarcă din nou bufferul EDIT din memorie.
ND	șterge N caractere consecutive.
E	termină editarea liniei și o înlocuiește.
NFX	caută al N-lea caracter X din linie și păstrează pointerul înaintea caracterului.
H	șterge tot ce este la dreapta pointerului și intră în modul de inserare.
I	intră în modul de inserare și inserează toate caracterele ce se introduc pînă cînd se tastează CR (return) sau ESC (SHIFT-E).
NKX	șterge caracterele de la pointer pînă la al N-lea caracter X (acesta nu va fi șters).
L	listează linia
Q	abandonează editarea fără a înlocui linia.
NR	înlocuirea a N caractere următoare cu N caractere introduse.
X	duce pointerul la sfîrșitul liniei și se intră în modul de inserare.
BLANK	duce pointerul la dreapta.
RUBOUT	duce pointerul la stînga.
CR	sfîrșitul editării liniei
ESC	sfîrșit mod de inserare.

5.1.5. *Comanda NEW*

Comanda NEW realizează eliminarea din memorie a programului curent pentru a face posibilă introducerea unui nou program.

Formatul general al comenzii :

NEW

Exemplu :

NEW

Prin această comandă se elimină din memorie orice informaţie legată de programul curent dacă acesta a existat, pregătindu-se astfel introducerea unui nou program.

5.1.6. *Comanda RENUMBER*

Comanda RENUMBER serveşte la renumerotarea liniilor BASIC. Formatul general al comenzii :

RENUMBER [<număr i>, <număr p>, <număr b>]

Unde :

<număr i> reprezintă numărul de linie de la care se începe renumerotarea.

<număr p> reprezintă mărimea pasului dintre numerele de linie atribuite la două linii consecutive.

<număr b> reprezintă numărul de linie minim ce se renumerotează.

5.1.7. *Comanda NULL*

Sintaxa comenzii este :

NULL <expr.>, <cod>

Unde <expr> este numărul caracterelor ce vor fi adăugate caracterelor CR şi LF la orice cerere de I/E de pe terminal ; <cod> este codul ASCII al caracterului. Pentru terminale lente este indicată execuţia comenzii la începerea sesiunii de lucru :

NULL 3,255

Comanda NULL ,O restabileşte configuraţia de bază.

5.1.8. *Comanda AUTO*

Cu comanda AUTO numerele de linie ale întregului program BASIC sînt generate automat. Formatul general al comenzii este :

AUTO [<nr. lin.> [, <pas>]]

Unde :

<nr. lin.> reprezintă numărul de linie atribuit primei linii din program. Valoarea implicită este 10.

<pas> reprezintă incrementul dintre două numere de linie consecutive. Valoarea implicită este 10.

5.1.9. *Comanda CLEAR*

Pentru a putea şterge toate variabilele, BASIC-PRAE dispune de comanda CLEAR. Formatul general al comenzii este :

CLEAR <număr>

Dacă argumentul număr este prezent, sealocă pentru zona de şiruri lungimea număr.

5.2. Comenzi de execuţie

În categoria comenzilor de execuţie intră comenzile de lansare a programului, de pregătirea relansării şi de continuare a unui program întrerupt. Aceste comenzi afectează direct execuţia unui program.

5.2.1. *Comanda CONTINUE*

Comanda CONTINUE permite reluarea execuţiei programului din punctul de întrerupere realizat prin instrucţiunea STOP.

Formatul general al comenzii:

CONTINUE

Împreună cu instrucţiunea STOP şi cu modul de lucru imediat, constituie mijloace foarte eficiente de punere la punct a programului.

Comanda CONTINUE nu poate înlocui comanda RUN şi nu are sens decît în punctele de întrerupere ale unui program lansat prin RUN.

5.2.2. *Comanda RUN*

Comanda RUN realizează lansarea în execuţie a programului curent în memorie.

Formatul general al comenzii:

RUN

În cazul unei comenzi RUN se lansează în execuţie programul curent dispus în memoria internă a utilizatorului de la instrucţiunea cu cel mai mic număr de linie.

Comanda RUN poate fi lansată şi dintr-un program dacă ea are număr de linie:

<nr. linie> RUN

5.3. Comenzi de bibliotecă

În categoria comenzilor de bibliotecă intră comenzile de salvare a programelor din memoria internă pe un suport extern, cele de încărcare a programelor dispuse pe un suport extern în memoria internă şi cele care asigură gestiunea programelor pe suportul extern.

5.3.1. *Comanda LOAD, ALOAD*

Comanda LOAD serveşte la încărcarea unui program de pe un suport extern indicat, în memoria internă.

Formatul general al comenzilor:

LOAD <nume fişier> <adresă început>

respectiv

ALOAD <nume fişier>

unde nume fişier este un şir de caractere cuprinse între apostroafe iar adresa început este adresa la care se începe încărcarea.

Programele scrise în cod ASCII se pot încărca prin comanda ALOAD. Fiecare linie începe cu un număr de linie şi se termină cu caracterul CR (retur de car). Sfîrşitul se detectează fie prin codul CTRL/Z din text fie prin marcajul EOF.

Înainte de a încărca programul specificat, comanda ALOAD şterge programul vechi din memorie. În timpul căutării programului se listează pe consolă, numele tuturor programelor întîlnite. Un program salvat cu SAVE nu poate fi încărcat cu comanda ALOAD. Programul salvat cu ASAVE poate fi încărcat cu ALOAD.

5.3.2. *Comanda PRECISION*

Precizia de calcul a componentei BASIC-PRAE este stabilită la 11 cifre semnificative. Totuşi, dacă utilizatorul doreşte ca rezultatele să fie tipărite cu o altă precizie mai mică decît 11, poate folosi comanda PRECISION.

Formatul general al comenzii este:

PRECISION <număr>

Unde:

<număr> este un număr mai mic sau egal cu 11 şi reprezintă precizia de tipărire a tuturor numerelor. (Interior se lucrează în continuare cu 11 cifre semnificative).

5.3.3. *Comanda SAVE, ASAVE*

Comanda SAVE produce salvarea blocului de memorie curent dispus în memoria internă pe un suport extern.

Formatul general al comenzii:

SAVE <nume fişier> <expr. 1>, <expr. 2>

Unde:

<nume fişier> este un şir de caractere cuprins între apostroafe
<expr. 1> este adresa de la care se salvează
<expr. 2> este adresa pînă la care se salvează

La execuţia comenzii SAVE se creează pe suportul indicat un fişier cu numele conţinut în specificaţie.

Exemplu:

SAVE „D", 3000, 6000

Această comandă SAVE salvează programul din memorie într-un fişier cu numele D.

Comanda:

ASAVE <nume fişier>

converteşte programul curent aflat în memorie în cod ASCII pentru a permite salvarea programului şi îl salvează apoi pe suportul extern.

5.3.4. *Comanda KILL*

Prin comanda KILL se recîştigă zona alocată pentru liste şi matrici. Formatul general e

KILL <nume matrice>, <nume matrice> ...

Unde:

<nume matrice> este numele unui masiv sau liste.

5.3.5. *Comanda AMERGE*

Comanda AMERGE serveşte pentru încărcarea şi interclasarea unui program de pe caseta cu programul aflat în memorie.

Sintaxa comenzii este:

AMERGE <nume fişier>

unde nume fişier este un şir de caractere cuprins între apostroafe.

În timpul căutării programului specificat se listează numele tuturor fişierelor întîlnite pe suport, astfel obţinîndu-se repertoriul programelor existente pe casetă.

Ciclul

algoritmi şi „pachete de programe"

PORTABILITATEA PROGRAMELOR DE CALCULE TEHNICO-ŞTIINŢIFICE

Dr. ing. *V. Sima*
I.T.C.I.

În contextul diversificării echipamentelor de calcul şi a creşterii continue a costurilor de elaborare a programelor, asigurarea portabilităţii reprezintă o cerinţă majoră. Lucrarea tratează cîteva aspecte ale problemei portabilităţii rutinelor de calcule tehnico-ştiinţifice. Sînt comentate şi exemplificate unele soluţii adoptate de autor în implementarea pe minicalculatoare a unor pachete de programe ştiinţifice, soluţii care pot fi folosite şi în realizarea sau adaptarea altor programe de acest gen.

1. PORTABILITATEA PROGRAMELOR ŞTIINŢIFICE

Ultimul deceniu a înregistrat, pe lîngă o creştere substanţială a volumului codurilor destinate calculelor tehnico-ştiinţifice, şi importante progrese calitative în acest domeniu. Pentru exemplificare, să considerăm biblioteca de subprograme Harwell (AERE Harwell, Anglia), care se bucură de prestigiu internaţional. Astfel, dacă versiunea în simplă precizie din anul 1972 a acestei biblioteci conţinea aproximativ 40 000 de linii sursă, fără comentarii, neindentate, greu de urmărit şi din care circa 15% erau scrise în limbajul de asamblare pentru IBM 360, în prezent biblioteca este extinsă semnificativ şi conţine coduri echivalente Fortran pentru rutinele scrise în asamblor şi versiuni portabile pentru majoritatea componentelor sale.

În ultimii ani au fost concepute şi realizate o serie de pachete de programe de calcule tehnico-ştiinţifice întrunind în mare măsură atributele de generalitate, eficienţă, *portabilitate*, precizie, robusteţe, comoditate în utilizare şi adaptare. Menţionăm cîteva dintre aceste pachete, disponibile şi în ţară : EISPACK — pentru rezolvarea problemelor algebrice de valori proprii, LINPACK — pentru analiza şi rezolvarea sistemelor de ecuaţii liniare, MINPACK — pentru minimizări de funcţii, IMSL şi SANDIA — ambele acoperind o gamă largă de calcule matematice şi statistice. Pachetele EISPACK, LINPACK şi MINPACK au fost realizate la Argonne National Laboratory, Argonne, IL, S.U.A. şi sînt distribuite, alături de IMSL şi de alte produse, de International Mathematical and Statistical Libraries (IMSL) Inc., Houston, TX, S.U.A. iar pachetul SANDIA a fost elaborat şi este distribuit de Sandia National Laboratories, Albuquerque, NM, S.U.A.

Problema portabilităţii a fost tratată în mod diferit de elaboratorii de programe. Astfel, în versiunea originală EISPACK [1] constantele dependente de maşina de calcul erau nedefinite, urmînd a fi precizate la fiecare implementare. În versiunea EISPACK integrată în pachetul SANDIA, valorile acestor parametri sînt comunicate prin intermediul unei rutine speciale. Pachetul LINPACK [2] este complet portabil. În schimb, IMSL Inc. dispune şi furnizează versiuni distincte ale bibliotecii IMSL pe diferite maşini din următoarele familii de calculatoare [3] : IBM 360/370, Xerox Sigma, Data General Eclipse, Digital Equipment (seriile 11, 10/20 şi VAX), Hewlett-Packard 3000 (seriile II şi III), Univac 1100, Honeywell 6000, Burroughs 6700/7700 şi CDC 6000/7000 şi Cyber 70/170.

Lucrarea prezintă unele soluţii utilizate de autor în realizarea unei *versiuni portabile* a bibliotecii de subprograme IMSL. Aceste soluţii au fost deja folosite, dar pot fi folosite şi în viitor, şi pentru elaborarea sau adaptarea altor pachete de programe de calcule tehnico-ştiinţifice. De asemenea, mostrele de coduri date în lucrare permit cunoaşterea stilului în care sînt scrise rutinele IMSL şi pot constitui modele de programare în Fortran demne de urmat. În plus, subprogramele originale listate în Anexele 2 şi 3 pot fi direct utilizate în alte programe.

Biblioteca IMSL cuprinde peste 500 de subprograme în simplă precizie și aproximativ 350 de subprograme în dublă precizie, scrise în limbajul Fortran IV standard și grupate pe capitole notate cu literele alfabetului. Adaptarea bibliotecii s-a făcut pe un minicalculator Coral 4011, pornind de la o variantă Felix (compilată cu opțiunea de „dublă lungime" DBL), furnizată de Oficiul de calcul al IRNE Pitești, variantă avînd la bază *versiunea în simplă precizie* pe calculatorul CDC Cyber 170/720. Dată fiind amploarea lucrării, trecerea pe minicalculator s-a făcut în cîteva etape, rezultînd mai multe biblioteci și pachete de programe. Dintre acestea menționăm bibliotecile: EIGAN (care implementează subrutinele din capitolul E, destinate analizei și rezolvării problemelor algebrice de valori și vectori proprii), TSPACK (analiza și predicția seriilor de timp — capitolul F) și VEMA (calcule cu matrice și vectori — capitolul V), și colecțiile de biblioteci: STAT (calcule statistice diverse — capitolele A, B, C, N, O, R, S, și parțial M și U) și MATH (calcule matematice diverse — capitolele D, E, G, I, L, V, Z, și parțial M și U). La realizarea versiunii în simplă precizie și-a adus contribuția și dr. ing. Th. D. Popescu (I.C.I.), care a adaptat și implementat subprogramele capitolelor B, C, D, F, I, R, S, și parțial M. De asemenea, pentru obținerea fișierelor în format exploatabil pe minicalculator au fost folosite programele de conversie elaborate de ing. Fl. Hartescu (I.C.I.)

Principalele dificultăți ivite în elaborarea variantei portabile a bibliotecii IMSL s-au datorat următoarelor elemente:

— prezența constantelor în cod;

— modul de memorare a cuvintelor întregi;

— calculul cu variabile complexe;

— calculul cu precizie extinsă.

Aceste probleme sînt tratate în secțiunile următoare ale lucrării. Menționăm că ultimele două aspecte, legate de realizarea versiunii în dublă precizie a bibliotecii pornind de la versiunea în simplă precizie, au implicat modificări însemnate în codul sursă. Astfel, lipsa unui compilator cu opțiune pentru calculul cu numere complexe în dublă precizie a necesitat codificarea în Fortran a unor operații aritmetice de bază. De asemenea, anumite calcule intermediare, care reclamă o precizie deosebită, au fost efectuate apelînd rutine Fortran speciale, cu precizie extinsă (aproape cvadruplă).

2. TRATAREA CONSTANTELOR DIN COD

Foarte multe programe de calcule tehnico-științifice necesită utilizarea unor *parametri dependenți de mașina de calcul* concretă, cum ar fi: cel mai mic (mare) număr reprezentabil, baza reprezentării în virgulă mobilă, precizia relativă a mașinii etc. Prezența explicită în cod a unor constante reprezentînd valorile acestor parametri reduce portabilitatea, făcînd necesară efectuarea de modificări la fiecare nouă implementare pe o altă mașină cu parametri diferiți. În biblioteca IMSL, acești parametri apar de regulă în instrucțiuni de inițializare DATA. Există însă și cazuri în care se utilizează variabile a căror valoare este fixată prin instrucțiuni de atribuire. Pentru depistarea acestor cazuri au trebuit studiate toate subprogramele, întrucît în general testele obișnuite în execuție nu detectează inadecvanța valorilor respective.

Soluția implementată pe minicalculator, care conferă maximum de portabilitate, a fost cea utilizată și în biblioteca SANDIA, și anume: parametrii dependenți de mașină sînt comunicați fiecărei rutine care îi reclamă utilizînd subprogramele funcție I1MACH, R1MACH sau D1MACH, adaptate după PORT Mathematical Subroutine Library (Bell Laboratories, S.U.A.). I1MACH, R1MACH și D1MACH furnizează valorile unor constante întregi, reale și, respectiv, dublă precizie, pentru următoarele familii de calculatoare: Burroughs 1700, 5700 și 6700/7700, CDC 6000/7000, Cray 1, Data General Eclipse, Harris 220, Honeywell 600/6000, IBM 360/370, Xerox Sigma 5/7/9, PDP 10 (cu procesor KA sau KI), PDP 11 și Univac 1100. Particularizarea acestor subprograme pentru o anumită mașină se face simplu prin înlocuirea cu blanc a caracterului C din prima coloană a liniilor comentariu corespunzătoare mașinii respective. I1MACH, R1MACH și D1MACH au un singur parametru, I, a cărui valoare precizează care anume constantă este cerută. Tabelele 1 și 2 prezintă semnificația valorilor funcțiilor I1MACH (I) și respectiv R1MACH (I) (sau D1MACH (I)).

Tabel 1

Semnificația valorilor funcției I1MACH (I)

I	I1MACH (I)	
1	unitatea de intrare standard	
2	unitatea de ieșire standard	
3	unitatea de perforare standard	
4	unitatea standard pentru mesaje de eroare	
5	numărul de biți pe unitatea de memorare (cuvînt) de tip întreg	
6	numărul de caractere pe cuvînt întreg	
7	a, baza reprezentării numerelor întregi	
8	s, numărul de cifre în baza a	
9	a^s—1, cel mai mare număr întreg reprezentabil	
10	b, baza reprezentării numerelor în virgulă mobilă	
11	t, numărul de cifre în baza b	pentru reprezentarea în simplă precizie
12	e_{min}, cel mai mic exponent	pentru reprezentarea în simplă precizie
13	e_{max}, cel mai mare exponent	pentru reprezentarea în simplă precizie
14	t, numărul de cifre în baza b	pentru reprezentarea în dublă precizie
15	e_{min}, cel mai mic exponent	pentru reprezentarea în dublă precizie
16	e_{max}, cel mai mare exponent	pentru reprezentarea în dublă precizie

Tabel 2

Semnificația valorilor funcțiilor R1MACH (I) și D1MACH (I)

I	R1MACH (I), D1MACH (I)
1	$b^{e_{min}-1}$, cel mai mic număr pozitiv reprezentabil
2	$b^{e_{max}}(1-b^{-t})$, cel mai mare număr reprezentabil
3	b^{-t}, cea mai mică spațiere relativă
4	b^{1-t}, cea mai mare spațiere relativă (precizia relativă)
5	$\log_{10}(b)$

Parametri dependenți de mașină apar în zeci de rutine din IMSL. În aceste rutine au fost suprimate instrucțiunile de inițializare sau de atribuire explicită și au fost incluse apeluri la subprogramele I1MACH, R1MACH sau D1MACH. Dificultăți mai mari am întîmpinat, de pildă, la adaptarea rutinelor destinate evaluării unor funcții speciale, matematice sau statistice (capitolul M), unde au trebuit identificate constante corespunzătoare valorilor ln (R1MACH (I)), pentru I=1, 2, 3, sau alte constante de acest gen.

Vom da acum cîteva detalii despre modificările făcute în cadrul bibliotecii EIGAN [4]. În rutinele EBALAC și EBALAF este apelată funcția I1MACH pentru a obține baza reprezentării în virgulă mobilă b. În cîteva cazuri a fost posibil să se elimine complet parametrii dependenți de mașină. De exemplu, s-a suprimat parametrul RDELP (sau REPS), care denotă precizia relativă a mașinii, din codurile rutinelor EHCUSS, ELRH1C și EQRT2S, prin înlocuirea instrucțiunilor de forma

```
IF (TEST .LE. RDELP*TEST1) GO TO ...
```

cu instrucțiuni

```
IF (TEST1+TEST .EQ. TEST) GO TO ...
```

Trebuie remarcat că cele două tipuri de instrucțiuni nu sînt întotdeauna echivalente. Astfel, pe mașinile avînd aritmetică cu *rotunjire corectă* în simplă precizie obținem 1.+ALPHA .NE. 1., pentru ALPHA $\geq$ EPS $\triangleq$ 1/2R1MACH (4) și 1.+ALPHA=1., pentru ALPHA<EPS. Instrucțiuni de al doilea tip sînt folosite și în pachetul LINPACK [2], ceea ce a permis ca acest pachet să nu conțină nici o constantă dependentă de mașină. Modificări similare au fost efectuate și în alte subrutine din EIGAN (anume ELRH2C, EQRH3F, EQRT1S și EQRT3S). Totuși, în aceste rutine, ca și în subrutinele EBNDV, EQRH1F și EQRT2F, variabila RDELP a fost reținută (dar furnizată de funcțiile R1MACH, respectiv D1MACH), căci apare ca numitor în unele situații rare care altfel ar implica împărțiri prin zero. De asemenea, în subrutinele de interes general (EIGBS, EIGCC, EIGCH, EIGRF, EIGRS, EIGZC și EIGZF), parametrul RDELP este utilizat în evaluarea indicelui de performanță.

În afara parametrilor dependenţi de maşină, în coduri pot apare unele *constante corespunzătoare algoritmului* implementat. Se recomandă ca la elaborarea programelor, constantele înl virgulă mobilă să fie definite prin instrucţiuni de iniţializare sau, eventual, prin instrucţiun de atribuire grupate în prima secţiune a codului, după declaraţiile de dimensiune şi tip. Acest lucru înlesneşte obţinerea unui cod portabil şi permite elaborarea comodă a versiunii în dublă precizie. Trebuie evitată utilizarea explicită a unor astfel de constante în instrucţiuni de calcul. Aceste recomandări au fost avute în vedere şi în cazul bibliotecii originale IMSL. Totuşi, în multe subprograme apar constante reale utilizate explicit. La realizarea variantei portabile a bibliotecii, cel puţin în acele rutine pentru care trebuia elaborată şi versiunea în dublă precizie am înlocuit constantele din cod cu variabile iniţializate prin DATA, completînd corespunzător declaraţiile de tip. Acest lucru a redus riscul ca în versiunea dublă precizie să rămînă constante în simplă precizie, asigurîndu-se portabilitate şi precizie sporite. (Menţionăm că unele compilatoare nu fac conversia de tip în expresii mixte, deci este uneori posibilă o pierdere substanţială de precizie.)

În codurile originale, constantele iniţializate prin DATA aveau maximum 14 zecimale (corespunzător versiunii în simplă precizie pentru calculatoarele CDC). Acolo unde a fost necesar şi cu putinţă, la elaborarea versiunii în dublă precizie am extins reprezentarea la *18 zecimale*. Astfel, pentru unele funcţii speciale am utilizat tabelele disponibile (de exemplu, pentru funcţia Erfc [5]). De asemenea, extinderea reprezentării s-a putut face uşor în cazul fracţiilor periodice. În alte cazuri, am preferat să calculăm efectiv valoarea constantei respective (de pildă, $(1.+17.^{1/2})/8.$), soluţie folosită şi în pachetul LINPACK [2].

O problemă colaterală care a apărut în faza iniţială de testare în execuţie a constituit-o apariţia unor erori fatale de „violare memorie“ pentru subrutine cu declaraţii de tipul

```
SUBROUTINE XX (TAU, ...)
REAL TAU (2, 1)
```

tabloul TAU putînd avea mai multe coloane şi fiind dimensionat corect în programul apelant. Eşecul se datora opţiunii implicite de *vectorizare* a tablourilor a compilatorului FOR, care în exemplul prezentat tratează TAU ca un tablou monodimensional cu 2 elemente. Am considerat secvenţa de mai sus portabilă, introducînd în bibliotecă modulul obiect generat fără vectorizare (opţiunea /NOVA).

3. UTILIZAREA TABLOURILOR ÎNTREGI

Probleme de portabilitate pot apare şi datorită variaţiei lungimii în biţi a cuvintelor întregi şi a caracterelor pentru diferite maşini. În tabelul 3 se prezintă valorile funcţiei I1MACH (I), pentru I=5, 6, 10 şi 11. I1MACH (5) şi I1MACH (6) reprezintă numărul de biţi şi, respectiv, numărul de caractere pe cuvînt întreg. Valorile pentru I1MACH (10) şi I1MACH (11) definesc baza b a reprezentării în virgulă mobilă şi, respectiv, numărul de cifre ale mantisei în baza b, pentru reprezentarea în simplă precizie. Cifrele din coloana "Cod" a tabelului indică numărul de biţi pe cuvînt real. Un cuvînt dublă precizie ocupă un spaţiu de memorie dublu.

Tabel 3

Valorile funcţiei I1MACH (I), I=5, 6, 10, 11

Nr crt.	Familia de calculatoare	Cod	I1MACH (I)			
			I=5	I=6	I=10	I=11
1	Burroughs 1700	H36	36	4	2	24
2	Burroughs 5700/6700/7700	H48	48	6	8	13
3	CDC 6000/7000	H60	60	10	2	48
4	Cray 1	H64	64	8	2	48
5	Data General Eclipse S/200	H32	16	2	16	6
6	Harris 220	H32	24	3	2	23
7	Honeywell 600/6000	H36	36	6	2	27
8	IBM 360/370	H32	32	4	16	6
9	Xerox Sigma 5/7/9	H32	32	4	16	6
10	PDP-10	H36	36	5	2	27
11	PDP-11	H32	16	2	2	24
12	Univac 1100	H36	36	6	2	27

Una din problemele de portabilitate, determinată de *inegalitatea cuvintelor real şi întreg*, apare atunci cînd un parametru formal declarat de tip întreg într-o rutină este tratat ca real (sau dublă precizie) în (sub)programul apelant, fiind totuşi necesară utilizarea valorilor întregi comunicate de rutină. Acesta este cazul subrutinei LINV3F din IMSL. Versiunea originală a acestei rutine dă rezultate eronate pe calculatoarele familiei PDP-11 sau pe minicalculatoarele româneşti. În Anexa 1 este listată în întregime versiunea în dublă precizie elaborată. Instrucţiunile fără numerotare în dreapta sînt cele adăugate pentru portabilitate. Listingul include şi comentariile, cu toate că sînt în limba engleză, căci acestea pot servi ca model de documentaţie minimală pentru un subprogram de calcule ştiinţifice.

Subrutina LINV3F operează asupra unei matrice A de ordin N, memorată într-un tablou de forma A(NA, N), şi eventual, asupra vectorului B de dimensiune N. În funcţie de opţiunea IJOB, se calculează: inversa matricei A (IJOB=1), soluţia ecuaţiei Ax=B (IJOB=2), soluţia ecuaţiei şi inversa (IJOB=3), şi determinantul (IJOB=4). În toate aceste cazuri se determină întîi o factorizare triunghiulară a unei matrice obţinută prin permutarea liniilor lui A, deci PA=LU, unde P este matricea de permutare, L este o matrice inferior triunghiulară cu elemente diagonale (nememorate) egale cu 1, iar U este o matrice superior triunghiulară. Această factorizare se obţine utilizînd subrutina LUDATF. Pentru rezolvarea sistemului Ax=B este apelată în continuare subrutina LUELMF. LUDATF necesită doi vectori de lucru de dimensiune N, unul întreg, IPVT, în care se înregistrează informaţii referitoare la permutările efectuate (care definesc matricea P), şi un vector real, EQUIL, al cărui conţinut nu prezintă importanţă în acest context. Informaţiile din vectorul IPVT sînt utilizate atît de LUELMF (pentru IJOB =2, sau 3), cît şi direct de LINV3F, pentru a obţine în final inversa $A^{-1}=U^{-1}L^{-1}P$ (pentru IJOB=1, sau 3). Spre a nu extinde lista de parametri ai subrutinei originale LINV3F, a fost însă prevăzut un singur tablou de lucru real WKAREA, de dimensiune 2N, care este folosit atît pentru IPVT, cît şi pentru EQUIL. Pe calculatoarele cu cod H32 şi cuvinte întregi pe 16 biţi, această soluţie eşuează, întrucît un element al vectorului WKAREA conţine două (sau patru) elemente ale vectorului IPVT în versiunea în simplă (dublă) precizie. Aparent, pentru versiunea în simplă precizie situaţia ar putea fi rezolvată simplu înlocuind în LUDATF şi LUELMF declaraţiile

```
        INTEGER    IPVT(1)
cu      REAL       IPVT(1)
```

dar aceasta nu ar corespunde documentaţiei şi ar implica efectuarea de modificări similare în toate (sub)programele care le apelează. De asemenea, să observăm că o declaraţie de tip INTEGER*4 ar fi rezolvat problema în simplă precizie pentru maşinile pe 16 biţi, dar nu ar fi fost operantă pe alte maşini, şi ar fi implicat modificări în alte rutine. În sfîrşit, o altă rezolvare posibilă ar fi fost adăugarea unui tablou întreg în lista de parametri ai rutinei LINV3F, cu actualizarea corespunzătoare a documentaţiei şi a tuturor subprogramelor care apelează LINV3F.

Soluţia portabilă implementată nu afectează interfaţa subrutinei LINV3F şi nu necesită nici o modificare în alte rutine. Se detectează automat cazul maşinilor cu lungime a cuvîntului întreg mai mică decît a cuvîntului real, comparînd I1MACH(5) cu MAX $\stackrel{\Delta}{=}$ max (I1MACH(10), I1MACH(11)). Conform Tabelului 3, dacă I1MACH(5)>MAX sîntem în situaţia standard, cu lungimi egale (STAN= .TRUE., în listing), exceptînd calculatorul Harris 220, pentru care oricum soluţia IMSL nu este posibilă. În caz contrar (STAN= .FALSE.), se ţine seama că WKAREA(J/2), J par (sau WKAREA(J/4), J multiplu de 4), conţine informaţii cu privire la permutările liniilor J—1 şi J (respectiv J—3, J—2, J—1 şi J), pentru versiunea în simplă (dublă) precizie. Soluţia este tehnic ceva mai complicată datorită faptului că permutările se aplică în ordine inversă. Menţionăm că modificările aduse nu afectează practic performanţele codului.

Ambele versiuni ale subrutinei LINV3F modificate au fost testate şi au funcţionat corect. De asemenea, au fost simulate pe minicalculator şi alte maşini. Spre exemplu, pentru calculatoarele din familiile FELIX C-256 sau IBM 360/370, s-a testat funcţionarea versiunii în simplă

precizie prin înlocuirea declarațiilor INTEGER IPVT(1) din rutinele LUDATF și LUELMF cu REAL IPVT(1), și prin folosirea unui cod ad-hoc pentru I1MACH :

```
INTEGER FUNCTION I1MACH(I)
IF(I.EQ.5) I1MACH=32
IF(I.EQ.10) I1MACH=16
IF(I.EQ.11) I1MACH=6
RETURN
END
```

Alte probleme de portabilitate au apărut la unele rutine utilitare, de afișare a rezultatelor. Vom ilustra acest lucru și vom indica o soluție posibilă portabilă în cazul cel mai simplu : afișarea unui vector folosind rutina USWFV. Primele declarații în codul original sînt :

```
SUBROUTINE USWFV(ITITLE, NC, A, N, INC, IOPT)
INTEGER    ITITLE(1), INC, IOPT, N, NC, NCA, NCMAX, NW
REAL       A(1)
DATA       NCMAX/20/
```

Subrutina permite afișarea elementelor 1,INC+1,2*INC+1, . . ., ale vectorului A, de dimensiune N, cu un format selectabil prin opțiunea IOPT, conform tabelului 4.

Tabel 4

Opțiuni pentru formatul de afișare (IOPT)

Opțiuni pentru 129 coloane	Opțiuni pentru 80 coloane	Format
1	2	F18.5
3	4	E15.6
5	6	E25.ISIG

ISIG din Tabelul 4 (pentru IOPT=5, sau 6) are o astfel de valoare încît să furnizeze o reprezentare cu precizie aproape completă a elementelor vectorului A. Valorile lui ISIG sînt precizate în Tabelul 5, conform documentației [3].

Tabel 5

Numărul de cifre zecimale afișate (ISIG)

ISIG	Precizie	Cod
7	simplă	H32
16	dublă	H32
9	simplă	H36
11	simplă	H48
14	simplă	H60

Rutina permite opțional afișarea unui titlu avînd NC⩽NCMAX=20 caractere. Dacă NC⩽0, acest titlu nu este afișat. Titlul este transmis prin tabloul întreg ITITLE, dar parametrul efectiv poate fi însă o constantă Hollerith, ca în exemplul de apel de mai jos :

```
CALL USWFV (10HTEST USWFV,10,A,15,1,6)
```

De remarcat că aici INC=1. O valoare INC≠1 poate fi folosită, de exemplu, pentru afișarea elementelor unei linii dintr-o matrice (INC>1), sau pentru afișarea elementelor unui vector în ordine inversă (INC=−1).

Apar astfel două probleme de portabilitate : *comunicarea şi afişarea titlului* şi *tipărirea valorilor la precizia maşinii* (IOPT=5, 6). Versiunea originală a rutinei USWFV nu este portabilă. Pentru afişarea titlului s-a utilizat formatul 2A10, iar ISIG a fost fixat la 14 (valori pentru CDC). Pentru maşinile cu cod H32 trebuie însă folosit ISIG=7, în simplă precizie, iar formatul pentru titlu poate fi 10A2 (pentru minicalculatoarele pe 16 biţi), sau 5A4 (pentru FELIX C-256). Soluţia portabilă realizată implementează pentru cele două probleme menţionate formatele variabile IFM şi IFS, respectiv, iniţializate şi actualizate corespunzător în timpul execuţiei, utilizînd informaţiile I1MACH(5) şi I1MACH(6). Datorită limitărilor de spaţiu, nu este posibil să listăm întreaga rutină. Totuşi, considerăm instructiv să prezentăm structura codului modificat, insistînd asupra modificărilor efectuate.

```
C     URMĂTOARELE 20 DE LINII SÎNT ADĂUGATE PENTRU PORTABILITATE
      INTEGER  IFM(6),IFS(7),NUM(5),NUS(3),I1MACH,NBIT,NC2,NC5
      DATA     IFM/2H(1, 2HX,, 2H 4, 2HA , 2H5 , 1H)/
      DATA     IFS/2H(4, 2HX,, 2H5E, 2H25, 2H. , 2H7 , 1H)/
      DATA     NUM/2H 2, 2H 4, 2H 6, 2H 5, 2H10/
      DATA     NUS/2H 9, 2H11, 2H14/
C     PREGĂTIRE FORMAT E25.ISIG
      IF (IOPT .LT. 5) GO TO 1
      NBIT=I1MACH(5)
      IF (NBIT .LT. 36) GO TO 1
      NBIT=NUS(NBIT/12—2)
      IFS(6)=NBIT
    1 NCHAR=I1MACH(6)
C     PREGĂTIRE FORMAT AFIŞARE TITLU
      IF (NCHAR .EQ. 5 .OR. NC .LE. 0) GO TO 2
      NC2=NCHAR/2
      NC5=5—NC2
      IF (NCHAR .NE. 6) NC5=NC5+1
      IFM(3)=NUM(NC5)
      IFM(5)=NUM(NC2)
    2 CONTINUE
C     PREGĂTIRE ŞI AFIŞARE TITLU
      NCA=MIN0(NC,NCMAX)
      NW=(NCA—1)/NCHAR+1
      CALL UGETIO (1,NIN,NOUT)
C     URMĂTOAREA LINIE — FORMAT PORTABIL
      IF (NC .GT. 0) WRITE(NOUT, IFM) (ITITLE(I), I=1,NW)
      .
      .
C     AFIŞARE VECTOR
      GO TO (30,30,35,35,40,40), IOPT
      .
      .
      .
C     URMĂTOAREA LINIE — FORMAT PORTABIL
   40 WRITE(NOUT,IFS) (A(I),I=1,N,INC)
      .
      .
      .
```

Subrutina elaborată implică un număr minim de operații suplimentare față de versiunea originală, dar operează pe toate calculatoarele din Tabelul 3, exceptînd Cray 1 și Harris 220. Testarea completă a funcționalității s-a făcut prin simularea pe minicalculator a altor mașini. De exemplu, pentru a simula FELIX C-256, am înlocuit declarația INTEGER ITITLE(1) cu REAL ITITLE(1) și am utilizat rutina ad-hoc I1MACH listată mai jos

```
INTEGER FUNCTION I1MACH(I)
IF (I .EQ. 5) I1MACH=32
IF (I .EQ. 6) I1MACH=4
RETURN
END
```

4. CALCULUL CU NUMERE COMPLEXE

Elaborarea versiunii în dublă precizie a bibliotecii IMSL a implicat o serie de modificări substanțiale în coduri. Dintre modificările generale, efectuate în toate subprogramele, menționăm: schimbarea declarațiilor de tip REAL în DOUBLE PRECISION; adaptarea corespunzătoare, inclusiv completarea valorilor numerice din instrucțiunile DATA; schimbarea numelor funcțiilor intrinseci (ABS, SQRT, SIGN etc). Amploarea acestor acțiuni nu trebuie subestimată; de pildă, există și rutine în care lista variabilelor inițializate prin DATA se întinde pe două pagini de listing. După efectuarea modificărilor indicate, codurile au fost compilate cu opțiunea /LI: 2, pentru a putea analiza listele de variabile locale, subprogramele apelate și funcțiile intrinseci referite și pentru a stabili corectitudinea formală a textului sursă. Menționăm că există și cazuri în care dintr-un subprogram în dublă precizie este apelat unul în simplă precizie, de exemplu pentru a obține o anumită distribuție de probabilitate. Aceste cazuri au trebuit tratate cu atenție, pentru a se respecta cerințele referitoare la tipul variabilelor.

În afara modificărilor generale, într-o serie de rutine au fost necesare modificări speciale pentru a permite efectuarea anumitor *operații cu numere complexe*. Majoritatea compilatoarelor Fortran, și în particular și cele disponibile pe minicalculatoarele românești, nu pot opera cu numere complexe avînd părțile reală și imaginară reprezentate în dublă precizie. Excluzînd din prima versiune a bibliotecii în dublă precizie circa 10 subprograme, care ar fi trebuit practic rescrise complet, din analiza codurilor originale a rezultat că este suficientă realizarea a trei operații de bază, și anume:

— calculul valorii absolute a unui număr complex dublă precizie;
— calculul rădăcinii patrate a unui număr complex dublă precizie;
— împărțirea a două numere complexe dublă precizie.

Aceste trei operații au fost implementate în trei subprograme originale: DCABS, DCSQRT și, respectiv, DCDIV. DCABS este un subprogram de tip funcție, iar celelalte două sînt subrutine. Toate au la bază algoritmii ALGOL prezentați în [6] și au fost scrise în Fortran IV, pentru portabilitate. Dat fiind faptul că aceste subprograme pot fi utilizate și în alte programe de calcule tehnico-științifice, am considerat oportun să le listăm în Anexa 2 (fără comentarii, din motive de spațiu). Pentru apelare comodă, listele de parametri conțin separat părțile reale și imaginare ale numerelor complexe care intervin. (Soluția alternativă, de grupare a părților reale și imaginare în vectori cu cîte 2 componente, ar fi necesitat modificări mai ample în rutinele apelante, fără a aduce un spor de eficiență.) Considerînd numerele complexe x=(XR,XI), y=(YR,YI) și z=(ZR,ZI), DCABS, DCSQRT și DCDIV calculează $DCABS=|x|$, $y=x^{1/2}$ și, respectiv, $z=x/y$. De remarcat că în toate cele trei subprograme nu se lucrează direct cu parametrii formali, ci cu variabile locale, pentru eficiență mai mare — prin evitarea adresărilor indirecte — și pentru a permite suprascrierea rezultatelor peste date. De asemenea, aceste subprograme pot fi ușor transcrise într-un limbaj mașină, dacă o aplicație specială cere viteză de execuție mai mare și cod mai compact decît rezultă în Fortran.

Menționăm că DCABS, DCSQRT și DCDIV au fost folosite și în versiunea în dublă precizie a pachetului EIGAN [4], pentru determinarea vectorilor proprii ai matricelor generale, utilizînd algoritmul QR. Cu același scop, aceste subprograme au fost incluse și în biblioteca matematică generală MIMAT [7], disponibilă pe *microcalculatoarele* M18 și M118. Întrucît compilatoarele Fortran cu care sînt înzestrate microcalculatoarele românești nu posedă opțiunea de calcul cu numere complexe, nici măcar în simplă precizie am implementat și versiunile cores-

punzătoare în simplă precizie ale rutinelor prezentate mai sus, anume: CABS, CSQRT și CDIV. Testele efectuate au confirmat corectitudinea și eficiența tuturor implementărilor.

În final, pentru a ilustra posibilitățile oferite de utilizarea rutinelor DCABS, DCSQRT și DCDIV indicăm cîteva dintre modificările mai ample efectuate pentru elaborarea variantei în dublă precizie a subrutinei MDGCI, destinată determinării inversei unei funcții generale de distribuție a probabilității, fiind date ordonatele densității. Subrutina folosește interpolare liniară sau neliniară pe o rețea de puncte echidistante sau neechidistante. Este necesară rezolvarea unei ecuații algebrice de ordin 1, 2, 3, sau 4. Calculul cu numere complexe intervine la rezolvarea ecuațiilor cubice sau cvartice. Vom considera cazul ecuațiilor cvartice. În versiunea în simplă precizie, structura codului, incluzînd doar una din secvențele modificate, este următoarea:

```
      REAL        BB,CC,D,FOURTH,HALF,PRED,QRED,SIXTH,THIRD,
     1            THREE,TWNT7,ZER
      COMPLEX     AL,BT,Z
      DATA        FOURTH/.25E0/,HALF/.5E0/,THREE/3.0E0/,ZER/0.0E0/,
     1            SIXTH/.16666666666667E0/,
     2            THIRD/.33333333333333E0/,
     3            TWNT7/.03703703703704E0/
      .
      .
      .
      Z=CMPLX(—QRED*HALF,ZER)
      Z=Z+CSQRT(CMPLX(PRED*PRED*PRED*TWNT7+QRED*QRED*FOURTH,ZER))
      Z=CEXP(CMPLX(THIRD,ZER)*CLOG(Z))
      Z=Z—CMPLX(PRED,ZER)/(THREE*Z)+CMPLX(CC*SIXTH,ZER)
      AL=CSQRT(Z+Z+CMPLX(FOURTH*BB*BB—CC,ZER))
      BT=(BB*Z—CMPLX(D,ZER))/(AL+AL)
      .
      .
```

Funcțiile CMPLX, CSQRT și CEXP sînt tratate de compilatorul Fortran. De remarcat că este necesar să se lucreze cu numere complexe datorită posibilității ca partea reală a numărului complex din care se extrage radicalul prima oară să fie negativă.

Codul în dublă precizie elaborat, echivalent matematic cu cel prezentat mai sus, este listat în continuare.

```
      DOUBLE PRECISION  BB,CC,D,FOURTH,HALF,PRED,QRED,SIXTH,THIRD,
     1                  THREE,TWNT7,ZER
      DOUBLE PRECISION  ALI,ALR,BTI,BTR,ZI,ZR,ZZI,ZZR
C                         Z=(ZR,ZI), AL=(ALR,ALI), BT=(BTR,BTI)
      DATA              FOURTH/.25D0/,HALF/.5D0/,THREE/3.0D0/,ZER/0.0D0/,
     1                  SIXTH/.16666666666666666667D0/,
     2                  THIRD/.33333333333333333333D0/,
     3                  TWNT7/.03703703703703703703 7D0/
      .
      .
      .
      ZR=—QRED*HALF
      CALL DCSQRT  (PRED*PRED*PRED*TWNT7+QRED*QRED*FOURTH,ZER,ZZR,
     1ZZI)
      ZR=ZR+ZZR
      ZZR=DEXP(THIRD)
      ZR=ZZR*ZR
      ZI=ZZR*ZZI
      CALL DCDIV (PRED,ZER,THREE*ZR,THREE*ZI,ZZR,ZZI)
      ZR=ZR—ZZR+CC*SIXTH
      ZI=ZI—ZZI
      CALL DCSQRT (ZR+ZR+FOURTH*BB*BB—CC,ZI+ZI,ALR,ALI)
      CALL DCDIV (BB*ZR—D,BB*ZI,ALR+ALR,ALI+ALI,BTR,BTI)
```

5. CALCULUL CU PRECIZIE EXTINSĂ

Convenim să numim aritmetică sau reprezentare cu *lungime simplă* (LS) aritmetica şi respectiv reprezentarea în virgulă mobilă — simplă sau dublă precizie — disponibilă pe calculatorul folosit. Anumite calcule matematice, cum ar fi: determinarea produsului scalar a doi vectori cu dimensiune mare, sau evaluarea unor funcţii speciale, impun utilizarea unei precizii extinse faţa de cea oferită de aritmetica LS. În acest scop, au fost elaborate tehnici care permit descrierea şi realizarea reprezentării şi aritmeticii cu *lungime multiplă* în funcţie de reprezentarea şi aritmetica LS. De exemplu, un număr cu *lungime dublă* (LD) este exprimat prin suma a două numere LS, unul dintre ele fiind aproape neglijabil în precizia LS. În cele ce urmează vom considera doar calcule în aritmetica LD şi vom prezenta soluţii portabile pentru efectuarea unor operaţii aritmetice cu precizie extinsă. Pentru aceasta, sînt însă necesare unele elemente şi precizări preliminare.

Preliminarii. Să notăm cu R mulţimea numerelor în virgulă mobilă (LS) reprezentabile în calculator şi să presupunem că R este de forma

$$R = \{x \mid x = mb^e,\ |m| < M,\ -D < e < E\}, \tag{1}$$

unde m, b, e, M, D, şi E sînt numere întregi a căror semnificaţie rezultă. Neluarea în considerare a depăşirilor aritmetice echivalează cu a considera D şi E infinite. Fie $x, y \in R$ şi $*$ o operaţie aritmetică binară pe R (de exemplu $+, -, \cdot, /$). Notăm $fl(x*y)$ rezultatul obţinut cu calculatorul prin aplicarea operaţiei $*$ asupra numerelor x şi y (utilizînd aritmetica LS).

Spunem că operaţia în virgulă mobilă corespunzătoare lui $*$ (pe scurt, operaţia $*$) este *corectă* dacă pentru $\forall\ x, y \in R$, $fl(x*y)$ este fie cel mai mare element din R mai mic decît sau egal cu $x*y$, fie cel mai mic element din R mai mare decît sau egal cu $x*y$. Deci, $fl(x*y) \in \{a, b\}$, unde $a, b \in R$ şi $[a, b]$ este cel mai mic interval cu extremităţi în R astfel încît $x*y \in [a, b]$. Dacă $x*y \in R$, $fl(x*y)$ este rezultatul corect. Operaţia $*$ este *optimală* (sau *cu rotunjire corectă*) dacă pentru $\forall\ x, y \in R$, $fl(x*y)$ este elementul din R cel mai apropiat de $x*y$.

Adunarea în virgulă mobilă este *cu trunchiere corectă* dacă $fl(x+y) = fl(y+x)$ (comutativitate) şi, pentru $\forall\ x, y \in R$, cu $|x| \geqslant |y|$, $fl(x+y)$ este cel mai mare element din R mai mic decît sau egal cu $x+y$, dacă $y \geqslant 0$, sau cel mai mic element din R mai mare decît sau egal cu $x+y$, dacă $y < 0$. Aşadar, dacă $x+y \in R$, rezultatul este trunchiat în direcţia lui $-y$. Scăderea este cu trunchiere corectă dacă pentru $\forall\ x, y \in R$, avem $fl(x-y) = fl(x+y')$, unde $y' = -y$ şi adunarea este cu trunchiere corectă. Adunarea şi scăderea sînt *supercorecte* dacă pentru $\forall\ x, y \in R$, $fl(x \pm y)$ este obţinut fie prin rotunjire corectă, fie prin trunchiere corectă.

Notăm $z = fl(x*y)$ (deci $z \in R$) şi fie zz corecţia necesară astfel încît să fie satisfăcută *exact* relaţia

$$z + zz = x*y.$$

În anumite condiţii [8], se poate arăta că $zz \in R$ şi că zz este (aproape) neglijabil cu precizia maşinii în raport cu $x*y$ (deci cu $z+zz$).

Spunem că perechea (z,zz), $z \in R$, $zz \in R$, este un *număr aproape LD*, dacă

$$|zz| \leqslant |z + zz|\, C \cdot 2^{-t} \tag{2}$$

unde C este o constantă nu mult mai mare decît 1, iar t este numărul de cifre binare ale mantisei în reprezentarea LS. Dacă în (2), $C = 1/(1+2^{-t})$, perechea (z,zz) este un *număr LD*.

Operaţii exacte. Algoritmii pentru calculul cu precizie extinsă au la bază adunarea exactă şi înmulţirea exactă a două numere LS. De aceea, prezentăm în continuare proceduri pentru realizarea acestor operaţii şi condiţii suficiente pentru valabilitatea rezultatelor.

Procedura cea mai simplă pentru *adunarea exactă* a numerelor x şi y, $x, y \in R$, cu $|x| \geqslant |y|$, este următoarea

$$z = fl(x+y)$$
$$w = fl(z-x),\quad zz = fl(y-w). \tag{3}$$

Dacă $|x| < |y|$, rolurile lui x şi y se inversează. Seturi de condiţii suficiente pentru ca z şi zz obţinute să fie astfel încît $z + zz = x + y$ sînt, de exemplu, a) sau b) indicate mai jos:

a) În (1) $b = 2$ sau 3, M este multiplu de b; adunarea este optimală sau supercorectă, iar scăderea este corectă.

b) În (1) b şi M sînt oarecare ; adunarea este cu trunchiere corectă, iar scăderea este corectă în condiţiile enunţate, $z-x \in R$, $y-w \in R$, deci w şi zz sînt exacte ($w=z-x$, $zz=y-w$). De asemenea, dacă într-un sistem binar, adunarea este optimală, atunci (z, zz) este număr LD, deci zz satisface (2) cu $C=1/(1+2^{-t})$. Condiţiile nu sînt însă suficiente dacă la calculul lui zz apar depăşiri inferioare, ceea ce este posibil în sistemele în virgulă mobilă cu normalizare. Dacă adunarea şi scăderea sînt optimale, dar b şi M sînt oarecare, nu se poate garanta că $w=z-x$. În acest caz, soluţia se poate obţine cu o procedură mai complicată, introducînd un termen de corecţie pentru w :

$$z=fl\,(x+y),$$

$$w=fl\,(z-x),\quad z1=fl\,(y-w),$$

$$v=fl\,(z-w),\quad z2=fl\,(v-x),$$

$$zz=fl\,(z1-z2).$$

Relaţiile (3) se pot scrie echivalent

$$w=fl\,(x-z),\quad zz=fl\,(w-y)$$

astfel încît codul Fortran corespunzător procedurii simple pentru $|x| \geqslant |y|$, este

```
Z=X+Y
ZZ=X-Z+Y
```

unde variabila w nu trebuie explicitată, ţinînd seama de ordinea de evaluare a expresiilor aritmetice în Fortran.

În cazul *înmulţirii exacte* vom presupune, pentru simplitate, următorul set de condiţii suficiente :

c) În (1), $b=2$, $M=2^t$, $D=E=\infty$, deci R este de forma

$$R=R\,(t)=\{x|\ x=m2^e,\ |m|<2^t\}\ ; \tag{4}$$

adunarea şi scăderea sînt optimale, iar înmulţirea este corectă.

Procedura pentru înmulţire exactă conţine două etape. În prima etapă numerele x şi y sînt descompuse fiecare în cîte două numere cu „lungime jumătate"

$$x=hx+tx,\quad y=hy+ty,$$

iar în a doua etapă se efectuează produsul exact utilizînd hx, tx, hy, ty. Secvenţa calculelor este următoarea :

$$p=fl\,(x\cdot c),\ q=fl\,(x-p),\ hx=fl\,(q+p),\ tx=fl\,(x-hx)\ (=x-hx)$$

$$p=fl\,(y\cdot c),\ q=fl\,(y-p),\ hy=fl\,(q+p),\ ty=fl\,(y-hy)\ (=y-hy)$$

$$p=fl\,(hx\cdot hy),\ q=fl\,(hx\cdot ty+tx\cdot hy) \tag{5.a}$$

$$z=fl\,(p+q),\ zz=fl\,(p-z+q+tx\cdot ty) \tag{5.b}$$

unde $c=2^{t-[t/2]}+1$, [.] denotînd partea întreagă. Justificarea procedurii se bazează pe faptul că, prin alegerea lui c, $hx\cdot hy \in R\,(t)$, $hx\cdot ty$, $tx\cdot hy \in R\,(t-1)$ (şi au acelaşi exponent) şi $tx\cdot ty \in R\,(t-2)$. Ca atare, cu ipotezele din c) referitoare la operaţii, în (5.a) p şi q sînt exacte, deci $z+z1=p+q$, cu $z1=fl\,(p-z+q) \in R\,(t-1)$. De asemenea, în (5.b) $z1+tx\cdot ty \in R$, astfel încît rezultă $z+zz=x\cdot y$.

Se poate arăta că procedura furnizează un număr (z,zz) aproape LD, căci $|zz| \leqslant |x\cdot y|\,\tau 2^{-t}/(1+\tau 2^{-t})$, unde $\tau=2$ pentru t par şi $\tau=3$, în caz contrar. Rezultatul (z,zz) poate fi transformat într-un număr LD efectuînd în final o adunare exactă a lui z cu zz :

$$w=z,\ z=fl\,(w+zz),\ zz=fl\,(w-z+zz).$$

Aritmetica LD. Procedurile prezentate mai sus pot fi aplicate direct pentru a obţine *algoritmi de calcul* în aritmetica LD. Pentru simplitate, presupunem îndeplinite condiţiile c).

Pentru a calcula suma a două numere (aproape) LD se poate aplica

Procedura AD : (z,zz)=(x,xx)+(y,yy) (pentru $|x| \geq |y|$)

1) Se adună exact x cu y, obţinînd (r,rr) astfel încît $r+rr=x+y$.
2) Se corectează rr : $s=fl(rr+yy+xx)$, astfel încît $r+s$ aproximează (x,xx)+(y,yy).
3) Se adună exact r cu s, obţinînd (z,zz), astfel încît $z+zz=r+s$.

Pentru scădere se procedează similar. Pentru a calcula produsul a două numere (aproape) LD se poate utiliza

Procedura MUL : (z,zz)=(x,xx)·(y,yy)

1) Se înmulţesc exact x şi y, obţinînd (r,rr) aproape LD, cu $r+rr=x\cdot y$.
2) Se corectează rr : $s=fl(x\cdot yy+xx\cdot y+rr)$, astfel că $r+s$ aproximează (x,xx)·(y,yy).
3) Se adună exact r cu s, obţinînd (z,zz) astfel încît $z+zz=r+s$.

Analog se pot descrie procedurile pentru operaţiile de împărţire şi rădăcină patrată, implicînd de asemenea înmulţiri şi adunări exacte. În [8] sînt prezentate codurile ALGOL corespunzătoare operaţiilor +, —, ·, /, şi $\sqrt{}$. Esenţial pentru însemnătatea acestor proceduri este faptul că analiza erorii furnizează rezultate de forma

$$|E^*| \leq (|x+xx| * |y+yy|)\, C^*\, 2^{-2t} \tag{6}$$

unde E* denotă eroarea în efectuarea operaţiei *, iar C* este o constantă nu mult mai mare decît 1. (De exemplu, $C\cdot \leq 9+\tau$, cu τ definit anterior). Formula (6) arată că eroarea este comparabilă cu cea care ar fi produsă lucrînd efectiv cu un sistem LD, cu reprezentare R (2t) şi aritmetică proprie, deşi de fapt calculele se fac cu aritmetica LS. Tehnica este eficace atunci cînd sistemul LS adoptat este sistemul cu cea mai mare precizie disponibil pe maşina de calcul. Pentru maşinile cu opţiune de dublă precizie se pot obţine rezultate cu precizie cvadruplă. De notat că efectuarea operaţiilor nu necesită acumulator cu lungime dublă şi că implementarea algoritmilor se poate face eficient în limbaje de înalt nivel, asigurîndu-se astfel portabilitatea.

Aplicaţii şi exemple. Tehnicile prezentate au fost aplicate la obţinerea versiunilor în dublă precizie ale unor subprograme din biblioteca IMSL. După cum rezultă din documentaţie [3], în IMSL sînt prevăzute trei subrutine pentru calculul cu precizie extinsă, anume VXADD, VXMUL şi VXSTO, care sînt apelate de aproximativ 70 de subprograme. Întrucît versiunea originală CDC se livrează numai în simplă precizie, nu am avut la dispoziţie codurile sursă pentru aceste subrutine. Codurile originale elaborate sînt prezentate (fără comentarii) în Anexa 3. Listele de parametri din VXADD, VXMUL şi VXSTO sînt respectiv (A,ACC), (A,B, ACC) şi (ACC,D), unde A, B şi D sînt variabile în dublă precizie, iar ACC este un tablou cu două elemente în dublă precizie. ACC are rol de acumulator conţinînd reprezentarea LD a rezultatului unor operaţii efectuate. Astfel, dacă (z,zz) este un număr (aproape) LD, ACC(1) conţine z, iar ACC(2) conţine zz. Subrutina VXADD adună numărul A la acumulator. VXMUL adună produsul numerelor A şi B la conţinutul acumulatorului şi furnizează rezultatul tot în acumulator. În sfîrşit, VXSTO transformă numărul LD din acumulator într-un număr LS, obţinut în D.

Subrutina VXADD realizează operaţia (z,zz) : =(x,0)+(z,zz) prin particularizarea pentru xx=0 a procedurii AD, prezentată mai sus. Sînt tratate adecvat ambele posibilităţi: $|x| \geq |y|$ şi $|x| < |y|$.

Subrutina VXMUL realizează operaţia (z,zz) : =(x,0)·(y,0)+(z,zz), parcurgînd următoarele etape:

i) Se efectuează înmulţirea exactă a lui x cu y, rezultînd (v,vv) aproape LD.
ii) Se transformă (v,vv) într-un număr LD (u,uu), utilizînd adunarea exactă.
iii) Se calculează suma (u,uu)+(z,zz), utilizînd procedura AD.

Remarcăm că etapele i) şi ii) echivalează cu particularizarea procedurii MUL pentru xx=yy=0. Etapa i) a fost implementată folosind procedura ALGOL *mul12* din [8]. Constanta $c=2^{t-[t/2]}+1$, folosită pentru descompunerea factorilor, a fost iniţializată prin instrucţiunea DATA. Această soluţie nu este portabilă, dar am preferat-o pentru eficienţă, întrucît se reduce volumul codului şi nu este necesară evaluarea constantei la fiecare apel al rutinei, apeluri care întotdeauna se fac în ciclu. (Evident, codul ar fi putut fi complicat astfel încît evaluarea lui c să se facă numai la primul apel întîlnit în program.) De altfel, chiar algoritmul care stă la baza rutinei VXMUL impune satisfacerea unor condiţii care nu sînt general valabile.

Subrutina VXMUL elaborată diferă de cea din IMSL. Aşa cum rezultă din documentaţie [3], versiunea IMSL apelează de 4 ori rutina VXADD, în timp ce în versiunea listată în Anexa

3 acest lucru este evitat. Ca atare, estimăm o economie de timp de calcul unitate centrală echivalentă cu 12 operații de adunare și 4 apeluri de rutină.

Subrutina VXSTO realizează operația simplă d=z+zz, (z,zz) fiind numărul (aproape) (LD memorat în acumulator.

Pentru *testarea rutinelor* de calcul cu precizie extinsă s-au evaluat sumele

$$\sum_{i=1}^{1\,000} |a_i|, \quad \sum_{i=1}^{1\,000} a_i b_i$$

unde $a_i = vx_i$, $b_i = vy_i$, și x_i și y_i fiind numere generate aleator în intervalul (0,1), iar v=ds, unde d se modifică după fiecare 100 de eșantioane prin program, s fiind un factor de scală cu valoare fixată la o rulare. În diferite rulări am modificat inițializarea șirurilor aleatoare, iar s a avut valorile 1, 10, 100, 1 000, 1.E5, 1.E15, 1.E—10 etc. Valori pentru s prea mari, de exemplu s=1.E20 (sau prea mici, ca s=1.E—15), au produs depășiri aritmetice superioare (inferioare) explicabile. Problemele de test prezentate au fost rezolvate *în simplă precizie* cu trei programe diferite :

TEST1 — program în simplă precizie pură ;

TEST2 — program cu calcule intermediare în dublă precizie, folosind funcția DBLE a compilatorului Fortran ;

TEST — program apelînd rutinele VXADD, VXMUL și VXSTO *scrise în simplă precizie.* Nu am utilizat un program complet în dublă precizie pentru ca datele de intrare să poată fi identice. Din analiza comparativă a rezultatelor obținute au reieșit următoarele concluzii

○ În toate cazurile subrutina VXADD s-a comportat perfect, dovedindu-se echivalentă calculelor în dublă precizie. Într-adevăr, valorile afișate de programele TEST și TEST2 aveau 14—17 cifre zecimale semnificative identice, cu 7—9 cifre exacte în plus față de valorile afișate de TEST1. De notat că la testele preliminare VXADD a furnizat aparent doar 1—2 cifre exacte în plus față de TEST1. De fapt, valorile martor erau greșite, datorită prezenței în TEST2 a instrucțiunii TEMP=TEMP+DABS (A (I)), în loc de TEMP=TEMP+DABS (DBLE (A (I)).

○ În toate cazurile subrutina VXMUL s-a comportat foarte bine. Valorile afișate de programele TEST și TEST2 aveau 13—15 cifre zecimale identice, cu 6—8 cifre exacte în plus față de cele afișate de TEST1.

○ VXADD și VXMUL necesită aproximativ de 3,5 și, respectiv, de 4 ori mai mult timp unitate centrală decît utilizarea funcțiunii DBLE a compilatorului.

Aceste concluzii pot fi probabil extrapolate și în cazul versiunilor în dublă precizie ale rutinelor VXADD, VXMUL și VXSTO. Pentru a putea aprecia niște rezultate numerice și în acest caz ar fi fost însă necesare valori martor cu precizie cvadruplă.

Ținînd seama de consumul mai mare de timp implicat de utilizarea rutinelor cu precizie extinsă, în biblioteca VEMA [9], destinată operațiilor de bază cu matrice și vectori, pentru acele rutine care apelează VXADD, VXMUL și/sau VXSTO am elaborat și versiuni în care aceste apeluri sînt suprimate, care nu utilizează deci precizie extinsă. Această soluție permite mai mare flexibilitate, făcînd posibilă adaptarea codurilor la necesitățile problemelor concrete.

Menționăm că versiunile în simplă precizie pentru VXADD, VXMUL și VXSTO pot fi folosite pe acele calculatoare sau sub acele sisteme de operare care nu au acumulator pe dublu cuvînt, deci care nu tratează realmente opțiunea DBLE a compilatorului Fortran. Aceasta este, de pildă, cazul unor sisteme pe microcalculatoare.

Ca exemplu banal de utilizare a rutinelor de calcul cu precizie extinsă, prezentăm o rutină pentru evaluarea produsului scalar a doi vectori A și B, de lungime N.

```
      DOUBLE PRECISION FUNCTION DSCAL (A,B,N)
      DOUBLE PRECISION          A,ACC(2),B,ZERO
      DATA                ZERO/0.0D0/
      ACC(1)=ZERO
      ACC(2)=ZERO
      DO 1 I=1,N
    1 CALL VXMUL (A(I),B(I),ACC)
      CALL VXSTO (ACC, DSCAL)
      RETURN
      END
```

În încheiere, vom discuta succint cîteva exemple de utilizare directă în coduri a facilităţilor de *calcul cu precizie extinsă*. Acest lucru este esenţial în primul rînd la evaluarea unor funcţii matematice speciale. Astfel în rutina MMBSJ0 din IMSL, destinată evaluării funcţiei Bessel de speţa I şi ordin 0, se adoptă o reprezentare LD pentru constanta 2π şi se iau măsuri pentru a prezerva precizia, folosind instrucţiuni de calcul de forma

```
PROD=((AX—XO1)—XO2) * (AX+XO1)
```

unde (XO1,XO2) este un număr LD. La fel, în subrutina MMBSYN, destinată calculului funcţiei Bessel de speţa II cu ordin fracţionar nenegativ şi argumente pozitive, se utilizează o tehnică de sumare cu precizie extinsă, similară celei din VXADD, şi se introduce o variabilă suplimentară pentru a preveni o posibilă eroare de optimizare a codului, efectuată de compilator. Un exemplu simplu pentru păstrarea preciziei apare în rutina MMDELK (integrala eliptică completă de speţa II) ; pentru evaluarea expresiei $\eta=1-x^2$ se foloseşte următorul cod

```
Y=ABS(X)
ETA=(H—Y)+H
ETA=ETA+Y*ETA
```

unde H este iniţializat prin instrucţiunea DATA H/0.5D0/.

6. CONCLUZII

Asigurarea portabilităţii este una dintre principalele cerinţe impuse programelor de calcule tehnico-ştiinţifice. Rezolvarea eficientă a diferitelor probleme privind portabilitatea permite transferul comod şi sigur al programelor de la o familie sau categorie de calculatoare la alta. Datorită restricţiilor de spaţiu, în lucrare s-au făcut referiri doar la unele aspecte ale problematicii portabilităţii, şi anume : tratarea constantelor dependente de maşină şi algoritm, soluţionarea diferenţelor de reprezentare (mai precis, lungime) a întregilor, calcule cu numere complexe sau cu precizie extinsă. Au fost prezentate unele soluţii concrete, originale, utilizate de autor în implementări pe mini- şi microcalculatoare a unor pachete de programe valoroase, făcîndu-se referiri şi la rezultatele testelor efectuate, inclusiv prin simularea altor maşini. Aceste soluţii pot fi aplicate cu uşurinţă şi pentru realizarea sau adaptarea altor programe de calcule tehnico-ştiinţifice.

Mulţumiri. Autorul adresează mulţumiri ing. R. Popescu de la Oficiul de Calcul al IRNE Piteşti, şi dr. fiz. O. Petruş, de la Laboratorul de Fizică Computaţională, Universitatea Al.I. Cuza, Iaşi, pentru furnizarea la I.C.I. a unor versiuni FELIX C-256 a pachetelor de programe IMSL şi respectiv, SANDIA.

REFERINŢE BIBLIOGRAFICE

1. *Smith, B.T., J.M. Boyle, J.J. Dongarra, B.S. Garbow, Y. Ikebe, V.C. Klema, C.B. Moler*, **Matrix Eigensystem Routines – EISPACK Guide,** 2nd Ed., Lect. Notes in Comp. Sci., Vol. 6, Springer-Verlag, New York, 1976.
2. *Dongarra, J.J., C.B. Moler, J.R. Bunch, C.W. Stewart*, **LINPACK Users' Guide**, SIAM, Philadelphia, 1979.
3. * * * **IMSL LIBRARY. Reference Manual,** IMSL LIB-0008, IMSL Inc., Houston, 1980.
4. *Sima, V., Eigan.* General Description, **Rep. ICI TR-05.84,** 1984.
5. *Cody, W.J.*, Rational Chebyshev approximations for the error function, **Math. Comp.**, vol. 23, 107, pag. 631–638.
6. *Wilkinson, J.H., C. Reinsch*, **Handbook for Automatic Computation,** Vol. II Linear Algebra, Springer-Verlag, Heidelberg, 1971.
7. *Popescu, A., V. Sima, C. Nomolosanu*, **Mimat V2. Completarea bibliotecii matematice pe microcalculatorul MC-18,** Studiu I.C.I., 1984.
8. *Dekker, T.J.*, A floating-point technique for extending the available precision, **Numer. Math.**, Vol. 18, pag. 224–242, 1971.
9. *Sima, V., Vema.* General Description, **Rep. ICI, TR-06.84,** 1984.

ANEXA 1

LINV3F — SUBRUTINA DE INVERSARE A UNEI MATRICE GENERALE (VERSIUNEA ÎN DUBLĂ PRECIZIE)

```
C  MATH ROUTINE NAME — LINV3F                                          1
C                                                                      2
C  ................................................................... 3
C                                                                      4
C  COMPUTER          — PDP/DOUBLE                                      5
C                                                                      6
C  LATEST REVISION   — NOVEMBER 1, 1984                                7
C                      CENTRAL INST. FOR MANAGEMENT AND INFOR-
                       MATICS
C                                                                      9
C  PURPOSE           — IN PLACE INVERSE, EQUATION SOLUTION, AND/
                       OR                                             10
C                      DETERMINANT EVALUATION — FULL STORAGE
                       MODE                                           11
C                                                                     12
C  USAGE             — CALL LINV3F (A,B,IJOB,N,IA,D1,D2,WKAREA,IER)   13
C                                                                     14
C  ARGUMENTS A       — INPUT/OUTPUT MATRIX OF DIMENSION N BY
                       N. SEE                                         15
C                      PARAMETER IJOB.                                16
C            B       — INPUT/OUTPUT VECTOR OF LENGTH N WHEN
                       IJOB=                                          17
C                      2 OR 3. OTHERWISE, B IS NOT USED.              18
C                      ON INPUT, B CONTAINS THE RIGHT HAND
                       SIDE OF                                        19
C                      OF THE EQUATION AX=B.                          20
C                      ON OUTPUT, THE SOLUTION X REPLACES B.          21
C            IJOB    — INPUT OPTION PARAMETER. IJOB=I IMPLIES :       22
C                      I=1, INVERT MATRIX A. A IS REPLACED BY         23
C                       ITS INVERSE.                                  24
C                      I=2, SOLVE THE EQUATION AX=B. A IS             25
C                      REPLACED BY THE LU DECOMPOSITION OF A          26
C                      ROWWISE PERMUTATION OF A, WHERE U IS           27
C                      UPPER TRIANGULAR AND L IS LOWER                28
C                      TRIANGULAR WITH UNIT DIAGONAL.                 29
C                      THE UNIT DIAGONAL OF L IS NOT STORED.          30
C                      I=3, SOLVE AX=B AND INVERT MATRIX A.           31
C                       A IS REPLACED BY ITS INVERSE.                 32
C                      I=4, COMPUTE THE DETERMINANT OF A.             33
C                       A IS REPLACED BY THE LU DECOMPOSITION         34
C                       OF A ROWWISE PERMUTATION OF A.                35
C            N       — ORDER OF A. (INPUT)                            36
C            IA      — ROW DIMENSION OF MATRIX A EXACTLY AS           37
C                      SPECIFIED IN THE DIMENSION STATEMENT IN        38
C                      THE CALLING PROGRAM. (INPUT)                   39
C            D1      — INPUT/OUTPUT. IF THE D1 AND D2 COMPONENTS      40
C            D2        OF DETERMINANT(A)=D1*2**D2 ARE DESIRED,        41
C                      INPUT D1.GE.O. OTHERWISE, INPUT D1.LT.O.       42
C                      D2 IS NEVER INPUT.                             43
C            WKAREA  — WORK AREA OF LENGTH AT LEAST 2*N FOR IJOB      44
C                      =1 OR IJOB=3.                                  45
C                      WORK AREA OF LENGTH AT LEAST N FOR IJOB=       46
C                      = 2 OR IJOB=4.                                 47
```

```
C        IER          — ERROR PARAMETER. (OUTPUT)                           48
C                       WARNING WITH FIX                                    49
C                       IER=65 INDICATES THAT IJOB WAS LESS THAN            50
C                       1 OR GREATER THAN 4. IJOB IS ASSUMED TO             51
C                       BE 4.                                               52
C                       TERMINAL ERROR                                      53
C                       IER=130 INDICATES THAT MATRIX A IS                  54
C                       ALGORITHMICALLY SINGULAR. (SEE THE                  55
C                       CHAPTER PRELUDE).                                   56
C                                                                           57
C  PRECISION/HARDWARE   — SINGLE AND DOUBLE/H32                             58
C                       — DOUBLE/H36,H48,H60                                59
C                                                                           60
C  REQD. MATH ROUTINES — LUDATF,LUELMF,UERTST,UGETIO,I1MACH                 61
C                                                                           62
C-------------------------------------------------------------------------- 73
C                                                                           74
      SUBROUTINE LINV3F (A,B,IJOB,N,IA,D1,D2,WKAREA,IER)                    75
C                                                                           76
      DOUBLE PRECISION A(IA,1),B(1),WKAREA(1),C1,C2,D1,D2,WA,ZERO,          77
     *                 ONE,SUM,C,WJ
      REAL             W(2),WJP,WJP1
      INTEGER          IWP(2),NMOD
      LOGICAL          EVEN,STAN
C                               STAN IS.TRUE. FOR MACHINES WITH
C                                 INTEGER WORDS LENGTH EQUAL TO
C                                 REAL WORDS LENGTH
      EQUIVALENCE      (WJ,WJP1,W(1)),(WJP,JP1,IWP(1),W(2)),
     1                 (JP,IWP(2))
      DATA             ZERO/0.0D0/,ONE/1.0D0/                               80
C                                 FIRST EXECUTABLE STATEMENT                81
C                                 LU DECOMPOSITION OF A                     82
      CALL LUDATF (A,A,N,IA,0,C1,C2,WKAREA,WKAREA,WA,IER)                   83
      IF (D1 .LT. ZERO .AND. IJOB .GE. 1 .AND. IJOB .LT. 4) GO TO 5         84
      D1=C1                                                                 85
      D2=C2                                                                 86
    5 IF (IER .GE. 128) GO TO 60                                            87
      IF (IJOB .LE. 0 .OR. IJOB. GT. 4) GO TO 55                            88
C                                 SOLVE AX=B                                89
      IF (IJOB .EQ. 2 .OR. IJOB. EQ. 3) CALL LUELMF (A,B,WKAREA,N,IA,B)     90
      IF (IJOB .NE. 1 .AND. IJOB .NE. 3) GO TO 9005                         91
C                                 MATRIX INVERSION                          92
      A(N,N)=ONE/A(N,N)                                                     93
      NM1=N—1                                                               94
      IF (NM1 .LT. 1) GO TO 9005                                            95
      DO 40 II=1,NM1                                                        96
         L=N—II                                                             97
         M=L+1                                                              98
         DO 15 I=M,N                                                        99
            SUM = ZERO                                                      100
            DO 10 K=M,N                                                     101
               SUM = SUM—A(I,K)*A (K,L)                                     102
   10       CONTINUE                                                        103
            WKAREA(N+I) = SUM                                               104
   15    CONTINUE                                                           105
         DO 20 I=M,N                                                        106
            A(I,L)=WKAREA(N+I)                                              107
   20    CONTINUE                                                           108
         DO 30 J=L,N                                                        109
            SUM = ZERO                                                      110
```

```
          IF (J .EQ. L) SUM = ONE                                     111
          DO 25 K=M,N                                                 112
             SUM = SUM—A(L,K)*A(K,J)                                  113
   25     CONTINUE                                                    114
          WKAREA(N+J) = SUM/A(L,L)                                    115
   30   CONTINUE                                                      116
        DO 35 J=L,N                                                   117
          A(L,J)=WKAREA(N+J)                                          118
   35   CONTINUE                                                      119
   40 CONTINUE                                                        120
C                    PERMUTE COLUMNS OF A INVERSE                     121
      STAN = .TRUE.
      IF(I1MACH(5) .LE. MAX0 (I1MACH(10),I1MACH(11))) STAN = .FALSE.
      EVEN = .TRUE.
      IF(MOD(N,2) .NE. 0) EVEN = .FALSE.
      IF(STAN .AND. .NOT. EVEN) WJ = WKAREA((N+1)/2)
      IF(STAN) GO TO 41
        NMOD = MOD(N,4)
        IF (NMOD .EQ. 0) GO TO 41
           WJ = WKAREA((N+3)/4)
           IF(NMOD .LE. 2) WJP = WJP1
           IF(.NOT. EVEN) JP = JP1
   41 CONTINUE
      DO 50 I=1,N                                                     122
        J = N—I+1                                                     123
        IF (.NOT. STAN) GO TO 42
           IF(.NOT. EVEN) WJP = WJP1
           IF(EVEN) WJ = WKAREA(J/2)
           JP = JP1
           GO TO 44
   42   CONTINUE
        IF (MOD(J,4) .NE. 0) GO TO 43
           WJ = WKAREA(J/4)
           GO TO 44
   43   IF(EVEN) WJP = WJP1
   44   CONTINUE
        IF (J .EQ. JP) GO TO 47                                       125
        DO 45 K=1,N                                                   126
           C = A (K,JP)                                               127
           A(K,JP) = A(K,J)                                           128
           A(K,J) = C                                                 129
   45   CONTINUE                                                      130
   47   CONTINUE
        IF(STAN) GO TO 49
           IF(EVEN) JP = JP1
   49   EVEN = .NOT. EVEN
   50 CONTINUE                                                        131
      GO TO 9005                                                      132
   55 CONTINUE                                                        133
C                       WARNING WITH FIX — IJOB WAS SET               134
C                       INCORRECTLY                                   135
      IER = 65                                                        136
      GO TO 9000                                                      137
C                       TERMINAL ERROR — MATRIX A IS                  138
C                       ALGORITHMICALLY SINGULAR                      139
   60 IER = 130                                                       140
 9000 CONTINUE                                                        141
      CALL UERTST(IER,6HLINV3F)                                       142
 9005 RETURN                                                          143
      END                                                             144
```

ANEXA 2

DCABS, DCSQRT, DCDIV — SUBPROGRAME ÎN DUBLA PRECIZIE PENTRU OPERAȚII CU NUMERE COMPLEXE (MODUL, RĂDĂCINA PĂTRATA ȘI RAPORT)

```
      DOUBLE PRECISION FUNCTION DCABS(XR,XI)
C
C                                  SPECIFICATIONS FOR ARGUMENTS
      DOUBLE PRECISION XR,XI
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      DOUBLE PRECISION H,YR,YI,ONE,ZERO
      DOUBLE PRECISION DABSD,SQRT
      DATA             ONE,ZERO/1.0D0,0.0D0/
C                                  FIRST EXECUTABLE STATEMENT
      YR = DABS(XR)
      YI = DABS(XI)
      IF (YR .GE. YI) GO TO 5
      H = YR
      YR = YI
      YI = H
    5 DCABS = YR
      IF (YR .EQ. ZERO .OR. YI .EQ. ZERO) GO TI 10
      DCABS = YR * DSQRT (ONE + (YI/YR)**2)
   10 RETURN
      END
      SUBROUTINE DCSQRT(XR,XI,YR,YI)
C                                  SPECIFICATIONS FOR ARGUMENTS
C
      DOUBLE PRECISION XR,XI,YR,YI
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      DOUBLE PRECISION H,ZR,ZI,TWO,ZERO
      DOUBLE PRECISION DABS,DCABS,DSQRT
      DATA             TWO,ZERO/2.0D0,0.0D0/
C                                  FIRST EXECUTABLE STATEMENT
      H = DSQRT ((DABS(XR) + DCABS(XR,XI))/TWO)
      ZR = H
      ZI = XI
      IF (ZI .NE. ZERO) ZI=ZI/(TWO*H)
      IF (XR .GE. ZERO) GO TO 10
         IF (ZI .LT. ZERO) GO TO 5
            ZR = HI
            ZI = H
            GO TO 10
    5       ZR = —ZI
            ZI = —H
   10 YR = ZR
      YI = ZI
      RETURN
      END
      SUBROUTINE DCDIV(XR,XI,YR,YI,ZR,ZI)
C                                  SPECIFICATIONS FOR ARGUMENTS
      DOUBLE PRECISION XR,XI,YR,YI,ZR,ZI
C                                  SPECIFICATIONS FOR LOCAL VARIABLES
      DOUBLE PRECISION H,XXR,XXI,YYR,YYI,ZERO
      DOUBLE PRECISION DABS
      DATA             ZERO/0.0D0/
C                                  FIRST EXECUTABLE STATEMENT
```

```
      XXR = XR
      XXI = XI
      YYR = YR
      YYI = YI
      ZI = ZERO
      IF (YYR .NE. ZERO) ZR = XXR/YYR
      IF (XXI .EQ. ZERO .AND. YYI .EQ. ZERO) GO TO 20
      IF (DABS(YYR) .LE. DABS(YYI)) GO TO 10
        H=YYI/YYR
        YYR = H*YYI + YYR
        ZR = (XXR + H * XXI)/YYR
        ZI = (XXI — H * XXR)/YYR
        GO TO 20
C
   10 CONTINUE
        H = YYR/YYI
        YYI= H * YYR + YYI
        ZR = (H *XXR + XXI)/YYI
        ZI = (H *XXI — XXR)/YYI
C
   20 CONTINUE
      RETURN
      END
```

ANEXA 3

VXADD, VXMUL, VXSTO — SUBPROGRAME ÎN DUBLĂ PRECIZIE PENTRU CALCULE CU PRECIZIE EXTINSĂ (ADUNARE LA ACUMULATOR, ÎNMULŢITRE ŞI ACTUALIZARE ACUMULATOR, CONVERSIE REPREZENTARE)

```
      SUBROUTINE VXADD (A,ACC)
C                                  SPECIFICATIONS FOR ARGUMENTS
      DOUBLE PRECISION   A,ACC(2)
C                                  SPECIFICATIONS FOR LOCAL VARIA-
                                   BLES
      DOUBLE PRECISION   X,Y,YY,R,S
C                                  FIRST EXECUTABLE STATEMENT
      X=A
      Y = ACC(1)
      YY = ACC(2)
      R = X+Y
      IF (DABS(X) .LE. DABS(Y)) GO TO 10
        S = X — R + Y + YY
        GO TO 20
   10 CONTINUE
        S = Y — R + X + YY
   20 CONTINUE
      X = R + S
      ACC(1)= X
      ACC(2)= R — X + S
      RETURN
      END
      SUBROUTINE VXSTO (ACC,D)
```

```
C                                                SPECIFICATIONS FOR ARGUMENTS
      DOUBLE PRECISION D,ACC(2)
C                                                FIRST EXECUTABLE STATEMENT
      D = ACC(1) + ACC(2)
      RETURN
      END
      SUBROUTINE VXMUL  (A,B,ACC)
C                                                SPECIFICATIONSFOR ARGUMENTS
      DOUBLE PRECISION  A,B,ACC(2)
C                                                SPECIFICATIONS FOR LOCAL VARIABLES
      DOUBLE PRECISION   DABS
      DOUBLE PRECISION   CONST,HA,TA,HB,TB,P,Q,U,UU,Z,ZZ
      EQUIVALENCE        (HA,U),(TA,UU)
C                                                CONST IS A MACHINE DEPENDENT
C                                                  PARAMETER WHICH MUST BE INI-
C                                                  TIALIZED BY 2**(T-T/2)+1, WHERE
C                                                  T IS THE NUMBER OF MANTISSA
C                                                  DIGITS IN THE BINARY FLOATING-
C                                                  POINT SYSTEM USED.
C                                                  THE ALGORITHM DOES NOT
C                                                  DIRECTLY WORK FOR NON-BINARY
C                                                  SYSTEMS.
      DATA               CONST/268435457.0D0/
C                                                FIRST EXECUTABLE STATEMENT
      P = A * CONST
      HA = A — P + P
      TA = A — HA
      P = B * CONST
      HB = B — P + P
      TB = B — HB
      P = HA * HB
      Q = HA * TB + TA * HB
      Z = P + Q
      ZZ = P — Z + Q + TA * TB
      U = Z + ZZ
      UU = Z — U + ZZ
      Z = ACC(1)
      ZZ = ACC (2)
      P = Z + U
      IF (DABS(Z) .LE. DABS(U)) GO TO 10
         Q = Z — P + U + UU + ZZ
         GO TO 20
   10 CONTINUE
         Q = U — P + Z + ZZ + UU
   20 CONTINUE
      Z = P + Q
      ACC(1)=Z
      ACC(2) = P — Z + Q
      RETURN
      END
```

PACHET DE PROGRAME PENTRU MODELAREA ȘI PREDICȚIA SERIILOR DE TIMP

Dr. ing. *Th. D. Popescu*
I.T.C.I.

1. INTRODUCERE

Construcția modelelor dinamice stohastice ale seriilor de timp se practică într-o varietate de domenii : tehnică, economie, biologie, ecologie, astronomie, statistică aplicată etc. Aceste modele pot fi utilizate pentru :

- investigarea fenomenului ce a generat seria de timp ;
- predicția optimală a valorilor seriei ;
- determinarea relațiilor dintre două sau mai multe serii de timp ;
- stabilirea unor strategii de conducere optimală prin minimizarea efectului perturbațiilor asupra variabilelor dependente.

Problemele ce apar în acest domeniu sînt suficient de complexe pentru a justifica utilizarea tehnicii de calcul. Pachetele de programe specifice acestui domeniu, raportate în literatură, CAPTAIN (Shellswell, 1972), TIMSAC'74 (Akaike, Arahata, Ozaki, 1975), X-11-ARIMA (Dagum, 1979), ARIMAF(Goel, Rocco, 1982), PACK System (Reilly, 1984), AUTOB & J(Popescu, 1985a) etc. implementează diferite tehnici de modelare și predicție a seriilor de timp.

Pachetul de programe AUTOB & J care constituie obiectul acestei lucrări este destinat modelării și predicției seriilor de timp scalare și asigură suportul software pentru implementarea metodologiei Box-Jenkins de construcție a modelelor stohastice ARIMA și predicție optimală a valorilor seriei pe baza acestor modele. Metodologia de analiză Box-Jenkins reprezintă una dintre cele mai frecvent utilizate și mai precise tehnici de predicție pe termen scurt, fapt ilustrat și de atenția deosebită de care se bucură în ultimul timp în literatura de specialitate.

Pachetul de programe este complet interactiv și utilizarea sa necesită un minim de cunoștințe de informatică din partea utilizatorului. Produsul poate fi utilizat pentru :

- analiza și modelarea unor procese din industrie (chimie, metalurgie, energetică), transporturi, agricultură, ecologie, biologie, medicină, economie, asistență socială, etc.
- predicția evoluției unor parametri tehnologici : presiuni, temperaturi, concentrații, etc. din cadrul proceselor industriale sau a unor indicatori economici, vînzărilor, cererii de servicii publice etc.

Utilizarea produsului AUTOB & J prezintă următoarele avantaje :

- permite modelarea și predicția interactivă a seriilor de timp lăsînd mai mult timp utilizatorului pentru interpretarea rezultatelor ;
- reprezintă un instrument eficient de predicție, implementînd una dintre cele mai precise și mai larg utilizate tehnici de predicție pe termen scurt ;
- permite instruirea rapidă a celor interesați în domeniul modelării și predicției seriilor de timp utilizînd metodologia Box și Jenkins.

Pachetul de programe este implementat pe un minicalculator CORAL 4011 sub sistemul de operare RSX11M (facilități standard), este scris în Fortran'77 și utilizează biblioteca de subprograme TSPACK (Popescu, 1933) ce conține module software pentru modelarea și predicția seriilor de timp, abordarea Box-Jenkins.

După o scurtă trecere în revistă a metodologiei Box-Jenkins de modelare și predicție a seriilor de timp, în lucrare se prezintă structura și funcțiunile pachetului și procedura practică de utilizare a acestuia. În final este prezentată în detaliu o aplicație a produsului privind mode. larea și predicția unei serii de timp reprezentînd concentrația în cadrul unui proces chimic-

2. METODOLOGIA BOX — JENKINS DE MODELARE ŞI PREDICŢIE A SERIILOR DE TIMP

Fie o serie de timp nesezonieră {X}. Conform metodologiei de modelare Box-Jenkins (1976), evoluţia sa poate fi exprimată în termenii unei combinaţii a valorilor trecute ale seriei şi a unei secvenţe de tip zgomot alb, o serie de valori necorelate identic distribuite. În cadrul metodologiei se presupune că există un întreg (d), astfel încît staţionaritaea seriei

$$w_t = (1-B)^d X_t \tag{1}$$

este asigurată, unde $B^j x_t = x_{t-j}$. Prin urmare valorile seriei la momentul de timp **t** pot fi exprimate în felul următor:

$$w_t = \frac{\Theta_q (B)^q}{\Phi_p (B)^p} a_t \tag{2}$$

unde a_t este o secvenţă de tip zgomot alb şi :

$$\Theta_q (B)^q = 1 - \theta_1 B - \ldots - \theta_q B^q$$

$$\Phi_p (B)^p = 1 - \varphi_1 B - \ldots - \varphi_p B^p$$

Combinînd cele două ecuaţii de mai sus se obţine forma generală a modelului Box-Jenkins pentru serii de timp nesezoniere :

$$\Phi_p (B)^p (1-B)^d X_t = \theta_0 + \Phi_q (B)^q a_t \tag{3}$$

unde θ_0 este o constantă diferită de zero.

În notaţia Box-Jenkins, ecuaţia (3) reprezintă un model autoregresiv şi de medie alunecătoare integrat (ARIMA), (p, d, q), unde p este gradul părţii autoregresive a modelului (AR), d este ordinul de diferenţiere a seriei, iar q este gradul părţii de medie alunecătoare (MA) a modelului. Condiţia ca toate rădăcinile ecuaţiilor polinomiale în B : $\Phi(B)=0$ şi $\Theta(B)=0$, să se găsească în afara cercului unitate asigură staţionaritatea seriei w_t şi respectiv inversabilitatea modelului, garantînd că modelul astfel specificat este unic reprezentabil.

În cadrul abordării Box-Jenkins se parcurg de regulă următoarele etape : identificarea (specificarea) modelului, estimarea parametrilor modelului, verificarea modelului şi predicţia valorilor seriei, reprezentate în Fig. 1.

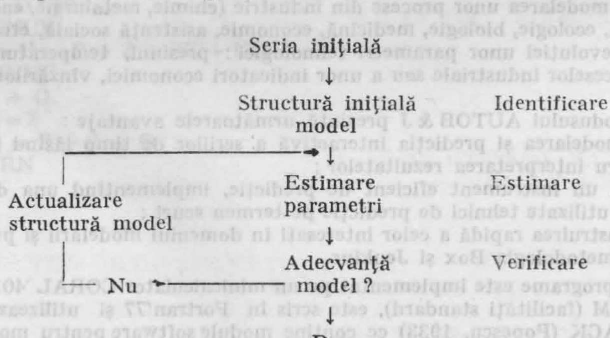


Fig. 1 Procesul de modelare şi predicţie

În etapa de identificare a modelului se pornește de la un set de valori ale parametrilor p, d, q ce se obține utilizînd o procedură ce va fi prezentată mai jos. În aceste condiții se determină un prim set de valori ale coeficienților modelului $\varphi_1, \varphi_2, \ldots, \varphi_p$ și $\theta_1, \theta_2, \ldots, \theta_q$. Modelul astfel obținut este verificat pentru a se stabili gradul său de adecvanță în raport cu datele seriei. Dacă în urma etapei de verificare se impune o nouă structură a modelului, ciclul de estimare a parametrilor modelului și verificarea acestuia se repetă pentru noul set de parametri p, d, q. Procedura continuă pînă la obținerea modelului reprezentativ al seriei, pe baza căruia se pot determina valorile de predicție ale seriei.

Principalele instrumente de calcul ce se utilizează în etapa de identificare a modelului seriei sînt funcțiile de autocorelație și de autocorelație parțială. Fie seria staționară w_t de medie u. Valoarea funcției de autocorelație de ordin k, în acest caz, reprezintă corelația dintre valorile w_t și w_{t-k}:

$$\rho_k = E\{(w_t - u)(w_{t-k} - u)\}/E\{(w_t - u)^2\} \quad \text{cu } \rho_{-k} = \rho_k$$

Valoarea funcției de autocorelație parțială de ordin k dintre w_t și w_{t-k}, φ_{kk}, reprezintă soluția sistemului:

$$\rho_j = \sum_{i=1}^{k} \varphi_{ki}\,\rho_{j-i}, \quad j = 1, 2, \ldots, k$$

În cazul în care $(1 - \varphi_1 B - \ldots - \varphi_k B^k)\, w_t = a_t$, $\varphi_{kk} = \varphi_k$ și $\varphi_{k+j,\,k+j} = 0$, pentru toate valorile $j \geqslant 1$.

Revenim acum la procedura de identificare a modelului:

(i) Dacă seria inițială $\{X_t\}$ nu este staționară, adică $d \neq 0$ în ecuația (3), funcția de autocorelație a seriei nu va descrește rapid pentru valori mari ale întîrzierii k, impunîndu-se în această situație diferențierea seriei în scopul obținerii staționarității acesteia.

(ii) Presupunem că în urma aplicării de un număr suficient de ori a operatorului de diferențiere 1—B, seriei $\{X_t\}$, am obținut o serie staționară $\{w_t\}$. În acest caz:

(a) Presupunînd q=0, deci un model autoregresiv de ordin p al seriei, ARIMA(p, d, o), valorile funcției de autocorelație a seriei vor descrește — pentru toate valorile k — conform ecuației:

$$\rho_k = \sum_{i=1}^{p} \varphi_i \rho_{k-i}$$

sau, cu alte cuvinte, ele se vor atenua sub forma unei unde exponențiale și/sau sinusoidale, în timp ce valorile funcției de autocorelație parțială φ_{kk} se vor anula pentru toate valorile $k > p$.

(b) Dacă însă p=0, deci se presupune un model de medie alunecătoare al seriei, ARIMA(0,d,q), structura funcției de autocorelație este de forma $\rho_k = 0$ pentru toate valorile $k > q$, în timp ce funcția de autocorelație parțială se atenuează sub forma unei combinații de unde exponențiale și sinusoidale.

(c) În cazul în care $p \neq 0$ și $q \neq 0$, deci seria admite un model ARIMA(p,d,q), structura funcției de autocorelație este descrisă de următoarea ecuație cu diferențe:

$$\rho_k = \sum_{i=1}^{p} \varphi_i \rho_{k-i}$$

pentru toate valorile întîrzierii $k > q$, ea atenuîndu-se sub forma unei combinații de unde exponențiale și sinusoidale.

Deoarece în situațiile practice nu cunoaștem structurile reale ale funcțiilor de autocorelație și de autocorelație parțială se impune estimarea acestora pe baza eșantioanelor seriei ce se analizează. Fie, de exemplu eșantioanele $w_1, w_2, \ldots$, aparținînd seriei globale $\{w_t\}$. În acest caz estimarea valorilor funcției ρ_k se poate obține pe baza următoarei relații:

$$r_k = 1/N \sum_{t=k+1}^{N} (w_t - \overline{w})(w_{t-k} - \overline{w}) / 1/N \sum_{t=1}^{N} (w_t - \overline{w})^2$$

unde N reprezintă numărul eșantioanelor seriei, iar $\overline{w}$ este media eșantioanelor seriei.

Valoarea estimată a funcţiei de autocorelaţie parţială φ_{kk}, va fi în acest caz $\hat{\varphi}_{kk}$ şi se obţine prin rezolvarea următorului sistem de ecuaţii :

$$r_j = \sum_{i=1}^{k} \varphi_{ki} r_{j-i} \quad j=1, \ldots, k$$

În cadrul analizei se va presupune că funcţiile de autocorelaţie şi de autocorelaţie parţială ale eşantioanelor disponibile ale seriei sînt identice cu cele ale seriei globale. Acest lucru este valabil pentru un număr suficient de mare de eşantioane, situaţie în care specificarea modelului seriei poate fi făcută corect.

Valorile estimate ale coeficienţilor modelului seriei (ecuaţia 3) reprezintă rezultatul minimizării sumei pătratelor reziduurilor Σa_t^2 utilizînd o tehnică ce va fi descrisă în continuare. Această tehnică necesită un prim set de valori ale parametrilor modelului pentru iniţializarea procedurii iterative de calcul, ce se determină în etapa de identificare a modelului pe baza valorilor funcţiei de autocorelaţie a seriei.

În ceea ce priveşte etapa de validare a modelului, Box şi Jenkins (1976) propun mai multe teste ce urmează a se efectua asupra reziduurilor modelului, generate în etapa de estimare a parametrilor modelului. Dacă notăm aceste reziduuri şi funcţia lor de autocorelaţie cu $\hat{a}_t$ şi respectiv cu $r_k(\hat{a})$ şi presupunem că seria reziduurilor reale $\{a_t\}$ este o secvenţă de tip zgomot alb, atunci $r_k(\hat{a})$ va avea valoarea medie nulă şi deviaţia standard egală cu aproximativ $1/\sqrt{N}$. Eşantioanele funcţiei de autocorelaţie a reziduurilor seriei reprezintă abateri de la comportarea tipică a unei secvenţe de tip zgomot alb a reziduurilor şi pot sugera chiar o alternativă de specificare a modelului seriei. Un test general, dar nu suficient de puternic, de punere în evidenţă a caracterului de zgomot alb a secvenţei reziduurilor, în acest caz, constă în compararea valorii Q şi a distribuţiei χ^2 pentru (M—p—q) grade de libertate, unde :

$$Q = N \sum_{k=1}^{M} r_k^2(\hat{a})$$

şi $M \geq 20$. În cadrul acestui test este posibilă utilizarea periodogramei cumulative a reziduurilor seriei (Box-Jenkins, 1976).

Presupunem că am obţinut un model acceptabil pentru seria $X_1, \ldots X_n$ şi urmărim determinarea valorilor de predicţie ale seriei X_{n+m} unde $m=1, 2, 3, \ldots$ etc. În plus se va presupune că dacă ne situăm la momentul (n), predicţia optimală — în termenii erorilor de predicţie minime — a valorii X_{n+m} reprezintă valoarea aşteptată condiţională la momentul (n). Totuşi, valorile condiţionale X_n, X_{n-1} etc. reprezintă valorile reale ale seriei, cunoscute la momentul (n), în timp ce valorile a_n, a_{n-1} etc. reprezintă reziduurile generate de modelul seriei ; valorile a_{n+1}, a_{n+2} etc. sînt egale cu zero în timp ce X_{n+1}, X_{n+2} etc. reprezintă valorile de predicţie ale seriei determinate la momentul (n). Metoda de predicţie, dacă se cunoaşte un model al seriei de forma ecuaţiei (3) reprezintă, prin urmare, un proces iterativ în care m ia valorile 1, 2, 3 etc., iar (n+m) se substituie variabilei (t) în ecuaţia modelului seriei.

Principiile referitoare la modelarea şi predicţia seriilor de timp sezoniere sînt legate de cele prezentate în cadrul acestei secţiuni, dar apar unele probleme legate de lipsa de informaţie asupra comportării teoretice a funcţiilor de autocorelaţie şi de autocorelaţie parţială.

3. STRUCTURA ŞI FUNCŢIUNILE PACHETULUI AUTOB&J

3.1. Structura pachetului

Pachetul de programe AUTOB&J este orientat pe comenzi. Comenzile sînt interactive şi solicită utilizatorului răspunsuri la întrebările formulate. Produsul este prietenos în raport cu utilizatorii (user friendly), complet interactiv, include o opţiune "help" şi verificarea răspunsurilor utilizatorului. De asemenea sînt incluse facilităţi de salvare a fişierelor — pot fi memorate modelul seriei, valorile reziduurilor, valorile de predicţie ale seriei etc. — la solicitarea utiliza-

torului. Aceste fişiere pot fi utilizate ca fişiere de date de intrare pentru acest pachet sau pentru alte pachete de programe. Structura sistemului de programe este reprezentată în Fig. 2.

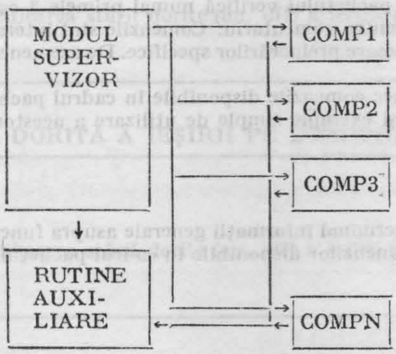


Fig. 2 Structura sistemului AUTOB&J

COMP1, . . . , COMPN reprezintă module program aplicative, corespunzătoare diferitelor funcţiuni ale pachetului, ce operează sub controlul unui modul supervizor al sistemului. Acesta din urmă lansează în execuţie modulele de program aplicative, specificate de utilizator prin comenzi, verifică disponibilitatea datelor necesare şi corectitudinea răspunsurilor utilizatorului la întrebările formulate de program.

Fiecare program de aplicaţie rezolvă un număr de probleme dintr-o anumită clasă, funcţie de opţiunile utilizatorului şi de datele furnizate. Programele permit un dialog întrebare-răspuns, adecvat unui utilizator începător sau sporadic.

3.2. Manipularea datelor

Pachetul AUTOB&J este orientat pe fişiere. Fiecare din seturile de date ce se analizează sau rezultatele furnizate de programele de aplicaţie : modelul seriei, valorile de predicţie etc. reprezintă fişiere pe disc ce pot fi regăsite prin specificarea numelor acestora.

Programul supervizor utilizează următorii vectori de date pentru :

- memorarea datelor originale ale seriei ORDAT ;
- memorarea datelor ce se analizează în mod curent ANDAT ;
- memorarea reziduurilor ce rezultă în urma determinării modelului seriei RESID ;
- memorarea valorilor rezultate în urma pedicţiei FOREC.

Comenzile utilizate pentru : încărcarea datelor seriei de pe disc în vectorul de date ORDAT, scrierea datelor din vectorul de date ANDAT într-un fişier de date pe disc, deschiderea unui fişier de date, închiderea unui fişier de date precum şi alte comenzi specifice manipulării datelor vor fi prezentate în cadrul unei alte secţiuni.

3.3. Funcţiunile pachetului

Pachetul AUTOB&J implementează toate etapele de calcul specifice metodologiei Box-Jenkins de analiză şi predicţie a seriilor de timp şi anume :

- transformarea datelor seriei, calculul diferenţelor şi diferenţelor sezoniere ale eşantioanelor seriei în scopul obţinerii staţionarităţii acesteia ;
- calculul funcţiilor de autocorelaţie şi de autocorelaţie parţială ale seriei în scopul identificării modelului ;
- estimarea preliminară a parametrilor părţii de autoregresie (AR) a modelului stohastic ARIMA ;
- estimarea preliminară a parametrilor părţii de medie alunecătoare (MA) a modelului stohastic ARIMA ;
- estimarea de verosimilitate maximă a parametrilor modelului stohastic ARIMA ;
- calculul valorilor de predicţie ale seriei şi al limitelor de probabilitate al acestora.

Aceste etape de calcul sînt acoperite de cele aproximativ 26 de comenzi ale pachetului ce se specifică de către utilizator.

Modulul supervizor al pachetului verifică numai primele 3 caractere ale comenzii, restul de caractere fiind utilizate pentru comentariu. Comenzile sînt interactive și solicită utilizatorului, prin întrebări, datele necesare prelucrărilor specifice. De asemenea se validează toate răspunsurile utilizatorului.

Prezentăm în continuare comenzile disponibile în cadrul pachetului, funcțiunile acestora, algoritmii utilizați, precum și exemple simple de utilizare a acestor comenzi.

Comanda HELP

Funcția : Afișează pe terminal informații generale asupra funcțiunilor pachetului, modului de organizare a datelor și comenzilor disponibile în cadrul pachetului.

Comanda INPUT

Funcția : Realizează citirea datelor seriei de timp dintr-un fișier de date de pe disc a cărui nume este specificat de către utilizator și încărcarea acestora în vectorii de lucru ORDAT și ANDAT. Fișierul de date de pe disc reprezintă o coloană de numere reale sau întregi.

Exemplu :

```
AB&J>INPUT
DATELE CITITE DIN FIȘIERUL ? ALFA
     120 DATE CITITE DIN FIȘIERUL : ALFA
          1.73406     7.45670     2.00543     1.98765 ....
```

Comanda OUTPUT

Funcția : Realizează scrierea datelor conținute în vectorul de date ANDAT într-un fișier pe disc al cărui nume este specificat de utilizator.

Exemplu :

```
AB&J>OUTPUT
FIȘIERUL ÎN CARE SE SCRIU DATELE ? BETA
     200 DATE SCRISE ÎN FIȘIERUL : BETA
```

Comanda OPEN

Funcția : Realizează deschiderea unui fișier pe disc.

Exemplu :

```
AB&J>OPEN
NUMELE FIȘIERULUI ? ALFA
```

Comanda CLOSE

Funcția : Realizează închiderea unui fișier pe disc.

Exemplu :

```
AB&J>CLOSE
```

Comanda FILE

Funcţia : Permite specificarea stării dorite (on, off) a ieşirii pe disc.

Exemplu :

```
AB&J>FILE
SPECIFICAŢI STAREA DORITĂ A IEŞIRII PE DISC (ON, OFF) ? ON
```

Comanda TERM

Funcţia : Permite specificarea stării dorite (on, off) a ieşirii pe terminal.

Exemplu :

```
AB&J>TERM
SPECIFICAŢI STAREA DORITĂ A IEŞIRII PE TERMINAL (ON, OFF) ? ON
```

Comanda PRINT

Funcţia : Realizează afişarea valorilor datelor seriei conţinute în vectorul ANDAT.

Exemplu :

```
AB&J>PRINT
     12 VALORI
          17.00000   16.60000   16.30000   16.10000
          17.10000   16.90000   16.80000   17.40000
          17.10000   17.00000   16.70000   17.40000
```

Comanda PLOT

Funcţia : Realizează reprezentarea grafică a datelor seriei ce se analizează.

Exemplu :

```
AB&J>PLOT
2.56277       5
2.25120      4 6
1.93962     3   7
1.62805¶  2      8
1.316481         90123456
1.00491                  78901234567
 .69333                             8901234567890
 .38176                                          123456789
        ..............^..........^..........^.........^..........^
                      10         20         30        40         50
```

Comanda PWIDTH

Fucţia : Realizează modificarea lăţimii reprezentării grafice a datelor seriei ; valoarea implicită este 66.

Exemplu :

```
AB&J>PWIDTH
LĂŢIMEA REPREZENTĂRII ? 50
```

Comanda PLENGTH

Funcţia : Realizează modificarea lungimii reprezentării grafice a datelor seriei ; valoarea implicită este 19.

Exemplu :

```
AB&J>PLENGTH
LUNGIMEA REPREZENTĂRII ? 10
```

Comanda ACF

Funcţia : Realizează calculul funcţiei de autocorelaţie a seriei ce se analizează.

Metoda : Pentru seria ce se analizează W_i, $i=1, \ldots, LW$, valoarea medie (AMEAN), dispersia (VAR) şi funcţia de autocorelaţie a seriei AC_j, $j=1, \ldots, K$ se determină cu următoarele relaţii :

$$AMEAN=1/LW \sum_{i=1}^{LW} W_i$$

$$VAR=1/LW \sum_{i=1}^{LW} (W_i - AMEAN)^2$$

$$ACV_j=1/LW \sum_{i=1}^{LW-j} (W_i - AMEAN)(W_{i+j} - AMEAN) \qquad j=1, \ldots, K$$

$$AC_j=ACV_j/VAR \qquad j=1, \ldots, K$$

Exemplu :

```
AB&J>ACF
NUMĂRUL TERMENILOR ? 10
VALOAREA MEDIE=0.00204                          DISPERSIA=0.13551
              -1.0  -0.8  -0.6  -0.4  -0.2  0.0   0.2   0.4   0.6   0.8   1.0
                +.....+.....+.....+.....+-----+.....+.....+.....+.....+.....+
 1 -0.411                  xxxxxxxxxxx
 2  0.016                             x
 3 -0.063                           xxx
 4 -0.013                             x
 5 -0.072                           xxx
 6 -0.011                             x
 7  0.139                             xxxx
 8 -0.066                           xxx
 9  0.038                             xx
10  0.017                             x
```

Comanda PACF

Funcţia : Realizează calculul funcţiei de autocorelaţie parţială a seriei.

Metoda : Calculul funcţiei de autocorelaţie parţială a seriei $PACV_j, j=1, \ldots, L$ utilizează valorile funcţiei de autocorelaţie determinate în cadrul comenzii ACF, astfel :

$$PACV_j = \begin{cases} AC_1 & j=1 \\ \dfrac{AC_j - \sum_{i=1}^{j-1} (AC_{j-i})(\varnothing_{j-1,i})}{1 - \sum_{i=1}^{j-1} (AC_i)(\varnothing_{j-1,i})} & j=2, \ldots, L \end{cases}$$

unde

$$\varnothing_{jk} = \begin{cases} PACV_j & \text{dacă } k=j \\ \varnothing_{j-1,k} - (PACV_j)(\varnothing_{j-1,j-k}) & k=1, 2, \ldots, j-1 \end{cases}$$

Exemplu :

```
AB&J>PACF
NUMĂRUL TERMENILOR ? 10
VALOAREA MEDIE=0.00204                      DISPERSIA=0.13551
            —1.0 —0.8 —0.6 —0.4 —0.2  0.0  0.2  0.4  0.6  0.8  1.0
              +-----+-----+-----+-----+----+----+----+----+----+----+
 1 —0.411                    XXXXXXXXXXX
 2 —0.184                          XXXXXX
 3 —0.163                           XXXXX
 4 —0.138                            XXXX
 5 —0.198                           XXXXX
 6 —0.207                           XXXXX
 7 —0.001                               X
 8  0.046                               XX
 9 —0.015                               X
10  0.037                               XX
```

Comanda TRANS

Funcţia : Realizează transformarea datelor seriei conţinute în vectorul ANDAT ; în urma operaţiei în vectorul ANDAT se vor găsi valorile rezultate.

Metoda : Se pot efectua următoarele tipuri de transformări :

(i) $W_i = Ln(W_i + SHIFT)$, $i=1, 2, \ldots, LW$. Constanta SHIFT se determină prin calcul astfel încît $\min (W_i + SHIFT) = 1$, $i=1, \ldots, LW$.

(ii) $W_i = W_i^{IP}$, $i=1, \ldots, LW$.

Exemplu :

```
AB&J>TRANS
AVERTISMENT : ACEASTĂ FUNCŢIE VA DISTRUGE CONŢINUTUL VECTO-
RULUI DATELOR SERIEI
EXPONENTUL TRANSFORMĂRII (SAU 0 ÎN CAZUL LOGARITMĂRII) ? 2
REALIZEAZĂ URMĂTOAREA TRANSFORMARE : DATA(I)=DATA(I)**2
```

Comanda DIFF

Funcţia : Realizează calculul diferenţelor sezoniere şi nesezoniere ale datelor seriei conţinute în vectorul ANDAT ; în urma operaţiei în vectorul ANDAT se vor găsi valorile rezultate.

Metoda : Pentru seria analizată W_i, $i=1, \ldots, LW$, diferenţele $Z_i = \nabla^{ID1} \nabla_{IS}^{ID2} W_i$, $=1, \ldots, LZ$ se determină cu relaţiile

$$\nabla W_i \equiv W_{i+1} - W_i$$

$$\nabla_{IS} W_i \equiv W_{i+IS} - W_i$$

$$\nabla_{IS}^{ID2} W_i \equiv \nabla (\nabla_{IS}^{ID2-1} W_i)$$

$$\nabla^{ID1} W_i \equiv \nabla (\nabla^{ID1-1} W_i)$$

unde

ID1 — ordinul diferenţei nesezoniere
ID2 — ordinul diferenţei sezoniere
IS — valoarea perioadei componentei sezoniere

Prin definiţie $\nabla^0 W_i \equiv W_i$, $\nabla_{IS}{}^0 W_i \equiv W_i$.

Exemplu :

```
AB&J>DIFF
AVERTISMENT : ACEASTĂ FUNCŢIE VA DISTRUGE CONŢINUTUL VEC-
TORULUI DATELOR SERIEI
ORDINUL DIFERENŢEI NESEZONIERE ? 1
ORDINIL DIFERENŢEI SEZONIERE ? 0
S-A DETERMINAT DIFERENŢA NESEZONIERĂ DE ORDIN 1
```

Comanda ARIMA

Funcţia : Realizează calculul estimaţiilor de verosimilitate maximă ale parametrilor modelului ARIMA şi reziduurilor modelului.

Metoda : Determinarea estimaţiilor de verosimilitate maximă ale parametrilor modelului ARIMA (p, d, q) :

$$\nabla^d X_t = \sum_{i=1}^{p} \varphi_i \nabla^d X_{t-i} + a_t - \sum_{i=1}^{q} \theta_i a_{t-i}$$

sau, într-o formă echivalentă :

$$X_t = \theta_0 + \sum_{i=1}^{p+d} \psi_i X_{t-i} + a_t - \sum_{i=1}^{q} \theta_i a_{t-i}$$

se poate face cu sau fără specificarea estimaţiilor preliminare ale parametrilor φ_i şi θ_i.

Funcţie de valoarea parametrului d seria iniţială X_t este transformată astfel :

$$W_t = \begin{cases} X_t - \overline{X}, & \text{dacă } d=0 \\ \nabla^d X_t, & \text{dacă } d>0 \end{cases}$$

unde $\overline{X}$ reprezintă valoarea medie a seriei.

În cazul nespecificării de către utilizator a valorilor estimaţiilor preliminare ale parametrilor modelului ARIMA, acestea se determină astfel :

a) Calculul parametrilor părţii AR a modelului

Într-o primă etapă se determină valorile funcţiei de autocovarianţă a seriei $\{W_t\}$: ACV_i, $i=1, \ldots, p+q+1$, parametrii părţii AR a modelului φ_i obţinîndu-se ca soluţie a următorului sistem liniar de ecuaţii :

$$Ax=b$$

unde

$$A_{ij}=ACV_{|q+i-j|+1} \quad i, j=1, \ldots, p$$
$$x_j=\varphi_j \quad j=1, \ldots, p$$
$$b_j=ACV_{q+j+1} \quad j=1, \ldots, p$$

Valoarea preliminară a parametrului θ_0 se determină cu relaţia :

$$\theta_0=\overline{X}\left(1-\sum_{i=1}^{q}\varphi_i\right)$$

b) Calculul parametrilor părţii MA a modelului

Evaluarea parametrilor θ_i ai modelului ARIMA al seriei se face cu relaţia :

$$\theta_i=-x_i/x_0, \quad i=1, \ldots, q$$

unde x_i şi x_0 reprezintă rădăcinile următorului sistem neliniar :

$$f_i(x_0, \ldots, x_q)=\sum_{j=0}^{q-i} x_j x_{i+j}-C_i'=0, \quad i=0, \ldots, q$$

Calculul rădăcinilor sistemului se realizează cu o metodă propusă de Brown (1969, 1971). Valorile C_i' reprezintă covarianţele modificate ale seriei şi se determină conform relaţiilor :

$$C_i'=\begin{cases}\sum_{j=0}^{p}\sum_{k=0}^{p}\varphi_j\varphi_k\, ACV_{|i+j-k|+1}, & p>0\\ ACV_{i+1}, & p=0\end{cases}$$

unde $\varphi_0=-1$. Dispersia reziduurilor modelului este x_0^2.

Dispunînd de valorile preliminare ale parametrilor modelului ARIMA al seriei, estimaţiile de verosimilitate maximă ale parametrilor se obţin prin minimizarea unei funcţii obiectiv de forma :

$$S\equiv\sum_{t=1}^{N}a_t^2,$$

cu $a_t=0$ pentru $t=1, \ldots, \max(p, q)$, utilizînd algoritmul modificat al pantei de coborîre maximă (Luenberger, 1973). Convergenţa algoritmului este presupusă în cazul îndeplinirii unuia din următoarele criterii :

(i) numărul de digiţi semnificativi fixat ai funcţiei obiectiv nu se modifică după un număr fixat de iteraţii ;

(ii) modificarea parametrilor modelului de la o iteraţie la alta este nesemnificativă.

Exemplu :

```
AB&J > ARIMA
SPECIFICAŢI STRUCTURA MODELULUI ? 0 0 2
DORIŢI SĂ INTRODUCEŢI ESTIMAŢIILE PRELIMINARE ALE PARAMETRI-
LOR ? N
PARAMETRI INIŢIALI MA :
      —.53388      .19593
ESTIMAŢIILE DE VEROSIMILITATE MAXIMĂ ALE PARAMETRILOR
MODELULUI ARIMA (0, 0, 2) PENTRU 120 DE VALORI ALE SERIEI
DUPĂ DIFERENŢIERE
    CONSTANTA DE MEDIE ALUNECĂTOARE =  —.13741
    DISPERSIA ZGOMOTULUI ALB =          .99076
    PARAMETRI MA :
      —.65530      .30809
```

Comanda AGAIN

Funcţia: Realizează calculul estimaţiilor de verosimilitate maximă ale parametrilor modelului ARIMA şi reziduurilor modelului utilizînd ca valori iniţiale ale parametrilor estimaţiile obţinute cu comanda ARIMA.

Metoda: Se utilizează procedura descrisă în cadrul comenzii ARIMA, pentru valori specificate ale estimaţiilor preliminare ale parametrilor modelulu i seriei.

Exemplu:

```
AB&J > AGAIN

PARAMETRI INIŢIALI AR :
  .90194

PARAMETRI INIŢIALI MA :
  .55885

ESTIMAŢIILE DE VEROSIMILITATE MAXIAMĂ ALE PARAMETRILOR
MODELULUI ARIMA (1, 0, 1) PENTRU 197 DE VALORI ALE SERIEI
DUPĂ DIFERENŢIERE
    CONSTANTA DE MEDIE ALUNECĂTOARE =     1.63077
    DISPERSIA ZGOMOTULUI ALB =     .09786

  PARAMETRI AR :
  .90442

  PARAMETRI MA :
  .56341
```

Comanda SMOD

Funcţia: Realizează salvarea modelului curent ARIMA (p; q; d; φ_i, $i=1, \ldots, p$; θ_i, $i=1, \ldots, q$; σ^2) într-un fişier pe disc al cărui nume este specificat de utilizator. σ^2 reprezintă dispersia reziduurilor.

Exemplu:

```
AB&J > SMOD

NUMELE FIŞIERULUI ? MOD1
MODELUL SALVAT ÎN FIŞIERUL : MOD1
```

Comanda RMOD

Funcţia: Realizează încărcarea în memorie a modelului ARIMA (p; q; d; φ_i, $i=1, \ldots, p$; θ_i, $i=1, \ldots, q$; σ^2) dintr-un fişier pe disc al cărui nume este specificat de utilizator.

Exemplu:

```
AB&J > RMOD

NUMELE FIŞIERULUI ? MOD1
MODELUL ÎNCĂRCAT DIN FIŞIERUL : MOD1
```

Comanda SWAP

Funcţia: Realizează interschimbarea datelor conţinute în vectorii de lucru ANDAT şi RESID în scopul analizei reziduurilor în cadrul etapei de validare a modelului.

Exemplu :

```
AB&J > SWAP
```

Comanda ORIG

Funcția : Transferă datele originale din vectorul ORDAT în vectorul ANDAT, în cazul în care prin utilizarea unor comenzi de tipul TRANS, DIFF etc. datele din vectorul ANDAT au fost modificate și se dorește a se lucra cu datele seriei originale.

Exemplu :

```
AB&J > ORIG
```

Comanda FOREC

Funcția : Realizează calculul valorilor de predicție și al limitelor de probabilitate al acestora, utilizînd ultimul model determinat al seriei.

Metoda : Calculul valorilor de predicție și al limitelor de probabilitate al acestora se realizează utilizînd modelul ARIMA al seriei care poate fi exprimat în termenii unei sume ponderate a valorilor curente și trecute ale zgomotului a_t :

$$X_t = \sum_{j=0}^{\infty} \psi_j a_{t-j}$$

unde $\psi_0=1$, iar ponderile ψ se obțin din ecuația :

$$\Phi(B)(1+\psi_1 B+\psi_2 B^2+\ldots)=\Theta(B)$$

Procedura de predicție este cea prezentată în secțiunea 2 a acestei lucrări, unde $a_n, a_{n-1}, \ldots$ reprezintă reziduurile generate de model, iar $a_{n+1}, a_{n+2}, \ldots$ etc. au valori nule ; $X_{n+1}, X_{n+2}, \ldots$ reprezintă valorile de predicție ale seriei. Predicția se realizează la momentul $t=n$.

Limitele de probabilitate $1-\varepsilon$, $X_{n+m}(-)$ și $X_{n+m}(+)$ pentru X_{n+m} sînt date de :

$$X_{n+m}(\pm)=\hat{X}_{n+m}\pm u_{\varepsilon/2}\left\{1+\sum_{j=1}^{m-1}\psi_j^2\right\}^{1/2} s_a$$

unde s_a^2 reprezintă valoarea estimată a dispersiei σ_a^2, iar $u_{\varepsilon/2}$ este un coeficient referitor la abaterea de la distribuția normală a zgomotului a_t.

Exemplu :

```
AB&J > FOREC

EPSILON ? 0,05
NUMĂR VALORI PREDICȚIE ? 8
```

MOMENT PREDICȚIE	PONDERI PSI	VALOARE PREDICȚIE	EROARE STD. 5.00 %
1	1.51587	7.27307	.72662
2	1.69071	6.94729	1.31954
3	1.64252	6.59124	1.80288
4	1.46333	6.24930	2.16213
5	1.22095	5.94715	2.40943
6	.96233	5.69674	2.56757
7	.71746	5.50060	2.66108
8	.50329	5.35531	2.71167

Comanda UPDATE

Funcţia : Realizează actualizarea valorilor de predicţie ale seriei după obţinerea une noi observaţii. În cadrul acestei comenzi se parcurg de regulă următoarele etape :

(a) Citirea datelor seriei şi încărcarea acestora în vectorii ANDAT şi ORDAT (ca în cazul comenzii INPUT) ;

(b) Încărcarea în memorie a modelului ARIMA al seriei pentru care se realizează predicţia (ca în cazul comenzii RMOD) ;

(c) Specificarea noii observaţii care se adaugă la datele seriei iniţiale ;

(d) Calculul valorilor de predicţie şi al limitelor de probabilitate al acestora (ca în cazu comenzii FOREC) :

(e) Salvarea într-un fişier pe disc a seriei actualizate.

Unele din aceste etape pot fi evitate dacă în locul introducerii numelui fişierului ce se solicită de către program se apasă tasta <CR>.

Metoda : Procedurile utilizate în cadrul acestei comenzi sînt cele prezentate în cadru comenzilor INPUT, RMOD, FOREC, înlănţuirea lor fiind realizată de secvenţa de calcul specifică comenzii UPDATE.

Exemplu :

```
AB&J > UPDATE

DATELE CITITE DIN FIŞIERUL : SER
     132 DATE CITITE DIN FIŞIERUL : SER
      3.7300          3.67000             3.77000            3.8300 ....

NUME FIŞIER MODEL ? MODX
MODELUL ÎNCĂRCAT DIN FIŞIERUL : MODX

ESTIMAŢIILE DE VEROSIMILITATE MAXIMĂ ALE PARAMETRILOR
MODELULUI ARIMA (2, 0, 0) PENTRU 132 DE VALORI ALE SERIEI
DUPĂ DIFERENŢIERE
     CONSTANTA DE MEDIE ALUNECĂTOARE =         .47593
     DISPERSIA ZGOMOTULUI ALB =        .13744

     PARAMETRI AR :
      1.51587        —.60716

VALOAREA NOII OBSERVAŢII ? 7.35

EPSILON ? .5
NUMĂR VALORI PREDICŢIE ? 8

MOMENT          PONDERI          VALOARE          EROARE STD.
PREDICŢIE       PSI              PREDICŢIE          5.00 %

      1          1.51587          7.06391           .72400
      2          1.69071          6.72130          1.31478
      3          1.64252          6.37566          1.79639
      4          1.46333          6.05972          2.15434
      5          1.22095          5.79066          2.40075
      6           .96233          5.57463          2.55832
      7           .71746          5.41051          2.65149
      8           .50329          5.29289          2.70190

FIŞIERUL ÎN CARE SE SCRIU DATELE ? BETA
     133 DATE SCRISE IN FIŞIERUL BETA
```

Comanda STAT

Funcţia : Realizează afişarea numărului de date conţinute în vectorii ORDAT, ANDAT, RESID, FOREC.

Exemplu :

```
AB&J > STAT

197 DATE ÎN VECTORUL DATELOR ORIGINALE ORDAT.
197 DATE ÎN VECTORUL DATELOR CE SE ANALIZEAZĂ ANDAT.
 75 Date ÎN VECTORUL REZIDUURILOR RESID.
  0 DATE ÎN VECTORUL VALORILOR DE PREDICŢIE FOREC.
```

Comanda CONC

Funcţia : Realizează concatenarea valorilor de predicţie la secvenţa datelor seriei analizate.

Exemplu :

```
AB&J > CONC
```

Comanda EXIT

Funcţia : Închide fişierul de ieşire pe disc, dacă acesta este deschis şi încheie sesiunea de lucru cu pachetul AUTOB&J.

Exemplu :

```
AB&J > EXIT
```

3.4. **Mesaje de eroare**

Practic este imposibil a se specifica acţiunea ce se impune din partea utilizatorului în cazul apariţiei unui mesaj de eroare, datorită numărului mare şi varietăţii mesajelor de eroare ce pot apare în timpul unei sesiuni de lucru. Totuşi, în general, acestea pot fi grupate în următoarele categorii :

(a) Mesaje de eroare ce apar în cadrul rutinelor de scriere/citire pe disc : eroare deschidere/închidere fişier, scriere/citire date etc.

(b) Mesaje de eroare datorate specificării de către utilizator a unor valori necorespunzătoare pentru parametrii de calcul ;

(c) Mesaje de eroare datorate unor probleme de natură numerică ce apar în procedurile de calcul incluse în pachet, de exp. lipsa convergenţei parametrilor modelului după un număr specificat de iteraţii etc.

Pentru mesajele de eroare aparţinînd primelor două categorii operaţia ce a generat apariţia erorii este reluată. În cazul mesajelor din ultima categorie, utilizatorul urmează să-şi reevalueze strategia aleasă.

3.5. **Restricţii**

Pachetul AUTOB&J este proiectat pentru :

(a) Numărul maxim de date al seriei = 500.

(b) Numărul maxim de valori ale funcţiilor de autocorelaţie şi de autocorelaţie parţială = 50.

(c) Ordinul maxim al părţii AR şi al părţii MA a modelului stohastic ARIMA = 20.

(d) Numărul maxim de valori de predicţie ale seriei = 100.

(e) Numărul maxim de iteraţii pentru estimarea de verosimilitate maximă a parametrilor modelului stohastic ARIMA = 25.

(f) Numărul de digiţi semnificativi ai funcţiei obiectiv utilizate în cadrul algoritmului de estimare prin metoda verosimilităţii maxime a parametrilor modelului ARIMA = 6.

În cazul în care aceste restricţii nu sînt respectate, utilizatorul va fi avertizat printr-un mesaj adecvat şi sesiunea de lucru continuă.

4. PROCEDURA PRACTICĂ DE MODELARE ŞI PREDICŢIE

În scopul facilitării înţelegerii modului de utilizare a pachetului de programe în cadrul unei probleme practice de modelare şi predicţie, în Fig. 3 se prezintă o diagramă ce ilustrează modul în care programele individuale, specificate prin comenzi, interacţionează unul cu celălalt pentru atingerea obiectivului final. În cadrul diagramei, programele incluse în aria mărginită de liniile întrerupte se utilizează în mod repetat, pînă cînd datele ce rezultă sînt aduse într-o formă adecvată prelucrării următoare.

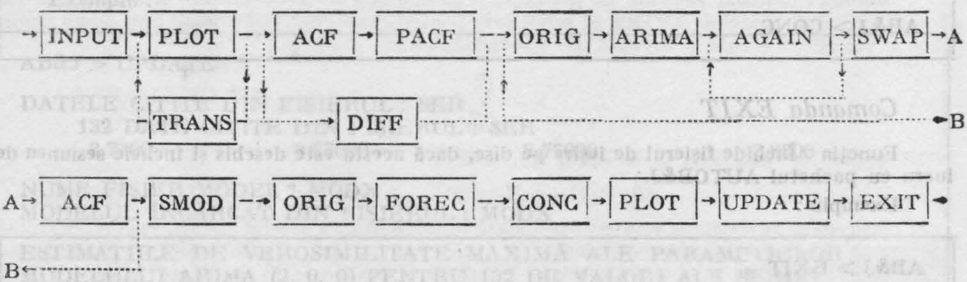


Fig. 3 Procedura practică de modelare şi predicţie a seriilor de timp utilizînd pachetul AUTOB&J

La începutul unei sesiuni de lucru şi după terminarea etapelor de calcul specifice unei comenzi, pe terminalul utilizator se va afişa mesajul "AB&J >", urmînd ca utilizatorul să specifice comanda dorită, conform strategiei de modelare şi predicţie alese. Comenzile sînt interactive şi adresează utilizatorului întrebările specifice etapelor de calcul din cadrul comenzi-specificate.

Prezentăm în continuare strategia de modelare şi predicţie, aşa cum apare în cadrul diagramei reprezentate în Fig. 3.

(i) Datele seriei sînt citite din fişierul specificat de utilizator (comanda INPUT) şi sînt reprezentate grafic (comanda PLOT). În cadrul etapelor următoare de modelare se pot utiliza datele originale ale seriei sau datele obţinute în urma unor transformări (comanda TRANS).

(ii) Se determină şi se analizează funcţiile de autocorelaţie şi de autocorelaţie parţială (comenzile ACF şi PACF) în scopul detectării unor eventuale nestaţionarităţi ale seriei sub forma unor tendinţe stohastice. În cazul prezenţei nestaţionarităţilor de acest tip, seria va fi diferenţiată de d ori ($d > 1$) pînă cînd se va observa că funcţia de autocorelaţie a seriei se atenuează rapid (se presupune că nestaţionaritatea se datorează celor d rădăcini egale cu unitatea ale polinomului Φ al modelului seriei). Fluctuaţiile sezoniere, de tipul ciclurilor în date, pot fi eliminate în mod similar prin calculul diferenţelor seriei cu o perioadă egală cu perioada fluctuaţiilor (comanda DIFF).

Dacă în continuare se dorește efectuarea analizei cu datele originale ale seriei, deoarece comanda DIFF distruge aceste date din vectorul de date ANDAT, se va utiliza comanda ORIG ce are drept scop reîncărcarea datelor originale ale seriei în vectorul de date ANDAT.

În cadrul etapei de analiză a funcțiilor de autocorelație și de autocorelație parțială pe lîngă valoarea parametrului d se vor determina de asemenea și ordinele părților AR și MA ale modelului stohastic ARIMA, p și q ; este posibilă chiar estimarea preliminară a parametrilor modelului.

(iii) În urma utilizării comenzii ARIMA se determină estimațiile de verosimilitate maximă ale parametrilor modelului ARIMA specificat în etapa anterioară (p, d, q) și valorile reziduurilor modelului. Dacă parametrii modelului seriei nu converg în mai puțin de 25 de iterații ale procedurii de estimare, se poate utiliza, în continuare, comanda AGAIN pentru reluarea procedurii de estimare a parametrilor modelului, cu valorile curente ale acestora, drept valori inițiale.

(iv) În scopul validării modelului seriei se va utiliza, într-o primă etapă, comnada SWAP care realizează încărcarea în vectorul de date ANDAT a seriei reziduurilor. În cazul în care rezultatele analizei reziduurilor (se poate utiliza și comanda ACF) nu denotă abateri ale acestora de la comportarea unei serii de tip zgomot alb, modelul ARIMA determinat este reprezentativ pentru seria analizată și poate fi salvat într-un fișier pe disc, utilizînd comanda SMOD. În caz contrar, procedura de analiză și modelare se va relua de la etapa de staționarizare a seriei și specificare a modelului.

(v) Pentru realizarea predicției valorilor seriei se utilizează într-o primă etapă, comanda ORIG în scopul încărcării în vectorul de date ANDAT a datelor originale ale seriei, după care calculul valorilor de predicție și al limitelor de probabilitate al acestora se efectuează cu comanda FOREC. Valorile de predicție pot fi concatenate cu datele seriei (comanda CONC) și reprezentate grafic (comanda PLOT).

(vi) În cazul în care se dorește actualizarea valorilor de predicție ale seriei, în urma obținerii unei noi observații, se va utiliza comanda UPDATE.

Deși toate programe de calcul incluse în pachet sînt eficiente din punct de vedere al calculelor efectuate, procedura completă de modelare și predicție necesită un oarecare timp de lucru ; aceasta în primul rînd datorită timpului de luare a unor decizii, necesar operatorului în primele etape ale analizei preliminare de specificare a modelului seriei, aceasta reprezentînd o artă și fiind necesară o anumită experiență pentru realizarea sa în mod eficient. Timpul total necesar rezolvării unei probleme de modelare și predicție se va reduce pe măsură ce utilizatorul va cîștiga experiență în recunoașterea unor forme tipice în cadrul reprezentărilor funcțiilor de autocorelație și de autocorelație parțială, care nu apar întotdeauna așa de clar cum sînt prezentate de Box și Jenkins (1976).

De asemenea, informația a priori de care se dispune, legată de natura fizică a procesului ce se analizează, poate contribui în mare măsură la reducerea timpului de analiză, în special în cadrul etapelor de specificare a modelului seriei, conducînd chiar la omiterea unor etape din cadrul analizei.

Cu toate că în cadrul acestei lucrări s-a căutat să se realizeze o prezentare netehnică a funcțiunilor pachetului, utilizarea unor comenzi necesită cunoașterea în detaliu a metodologiei de modelare și predicție.

5. APLICAȚIE

Pachetul de programe AUTOB&J a fost utilizat pentru modelarea și predicția unui număr mare de serii de timp reale. Cîteva din aceste aplicații sînt prezentate într-o lucrare elaborată de autor (Popescu, 1985b) și constau în modelarea și predicția unor serii de timp din procese industriale, modelarea unei populații biologice, modelarea și predicția transportului aerian internațional de călători.

În cele ce urmează se prezintă în detaliu rezultatele modelării și predicției unei serii de timp reprezentînd concentrația într-un proces chimic, măsurată la intervale de două ore, una din seriile din colecția de serii de timp utilizate ca exemple de test de către Box și Jenkins (1976). Seria conține 197 eșantioane. Aceeași serie este analizată și în lucrarea elaborată de Tertișco, Stoica, Popescu (1985) utilizînd analiza canonică de corelație.

Exemplul ce urmează ilustrează implementarea diferitelor etape, specifice metodologiei Box-Jenkins, în cadrul pachetului de programe AUTOB&J.

```
AUTOB&J VERSIUNEA 1.00

AB&J > INPUT

DATELE CITITE DIN FIŞIERUL ? BJS
     197 DATE CITITE DIN FIŞIERUL : BJS
        17.00000          16.60000          16.30000          16.10000...

AB&J > PRINT

     197 VALORI
        17.00000          16.6000           16.30000          16.10000
        17.10000          16.90000          16.80000          17.40000
        17.10000          17.00000          16.70000          17.40000
        ..............................................................
        ..............................................................
        ..............................................................
        17.40000          17.00000          18.00000          18.20000
        17.60000          17.80000          17.70000          17.20000
        17.40000

AB&J > ACF

NUMĂRUL TERMENILOR ? 20
VALOAREA MEDIE = 17.06294            DISPERSIA = 0.15837
     -1.0 -0.8 -0.6 -0.4 -0.2  0.0  0.2  0.4  0.6  0.8  1.0
       +    +    +    +    +    +    +    +    +    +    +
 1  0.572                      xxxxxxxxxxxxxxx
 2  0.496                      xxxxxxxxxxxxx
 3  0.400                      xxxxxxxxxxx
 4  0.356                      xxxxxxxxxx
 5  0.327                      xxxxxxxxx
 6  0.353                      xxxxxxxxxx
 7  0.390                      xxxxxxxxxxx
 8  0.321                      xxxxxxxxx
 9  0.303                      xxxxxxxxx
10  0.253                      xxxxxxx
11  0.188                      xxxxxx
12  0.164                      xxxxx
13  0.192                      xxxxxx
14  0.235                      xxxxxxx
15  0.138                      xxxx
16  0.178                      xxxxx
17  0.194                      xxxxxx
18  0.201                      xxxxxx
19  0.140                      xxxx
20  0.182                      xxxxxx
```

```
AB&J > PACF
NUMĂRUL TERMENILOR ? 20
VALOAREA MEDIE = 17.06294              DISPERSIA = 0.15837
          —1.0 —0.8 —0.6 —0.4 —0.2  0.0   0.2   0.4   0.6   0.8   1.0
            +....+....+....+....+....+....+....+....+....+....+
 1    0.572                          xxxxxxxxxxxxxx
 2    0.251                          xxxxxxx
 3    0.070                          xxx
 4    0.067                          xxx
 5    0.065                          xxx
 6    0.129                          xxxx
 7    0.150                          xxxxx
 8   —0.032                         xx
 9    0.009                          x
10   —0.020                          x
11   —0.069                        xxx
12   —0.020                          x
13    0.054                          xx
14    0.091                          xxx
15   —0.126                       xxxx
16    0.048                          xx
17    0.097                          xxx
18    0.069                          xxx
19   —0.071                        xxx
20    0.051                          xx
AB&J > DIFF
AVERTISMENT: ACEASTĂ FUNCŢIE VA DISTRUGE CONŢINUTUL VECTORULUI
DATELOR SERIEI
ORDINUL DIFERENŢEI NESEZONIERE ? 1
ORDINUL DIFERENŢEI SEZONIERE ? 0
S-A DETERMINAT DIFERENŢA NESEZONIERĂ DE ORDIN 1

AB& > ACF
NUMĂRUL TERMENILOR ? 20
VALOAREA MEDIE = 0.00204               Dispersia = 0.13551
          —1.0 —0.8 —0.6 —0.4 —0.2  0.0   0.2   0.4   0.6   0.8   1,0
            +....+....+....+....+....+....+....+....+....+....+
 1    0.411                     xxxxxxxxxxx
 2    0.016                          x
 3   —0.063                        xxx
 4   —0.013                          x
 5   —0.072                        xxx
 6   —0.011                          x
 7    0.139                          xxxx
 8   —0.066                        xxx
 9    0.038                          xx
10    0.017                          x
11   —0.045                          xx
12   —0.059                          xx
13   —0.017                          x
14    0.165                          xxxxx
15   —0.174                      xxxxx
16    0.035                          xx
17    0.012                          x
18    0.082                          xxx
19   —0.121                       xxxx
20    0.153                          xxxxx
```

```
AB&J > PACF

NUMĂRUL TERMENILOR ? 20
VALOAREA MEDIE = 0.00204              DISPERSIA = 0.13551
              —1.0 —0.8 —0.6 —0.4 —0.2  0.0   0.2   0.4   0.6   0.8   1.0

              +....+....+....+....+....+.....+.....+.....+.....+.....+

  1  —0.411             xxxxxxxxxxx
  2  —0.184                  xxxxxx
  3  —0.163                   xxxxx
  4  —0.138                    xxxx
  5  —0.198                  xxxxxx
  6  —0.207                  xxxxxx
  7  —0.001                       x
  8  —0.046                      xx
  9  —0.015                       x
 10   0.037                       xx
 11  —0.007                       x
 12  —0.070                     xxx
 13  —0.106                    xxxx
 14   0.106                       xxxx
 15  —0.087                     xxx
 16  —0.128                    xxxx
 17  —0.097                     xxx
 18   0.045                       xx
 19  —0.072                     xxx
 20   0.094                       xxx

AB&J > ORIG

AB&J > ARIMA

SPECIFICAŢI STRUCTURA MODELULUI ? 0 1 1

DORIŢI SĂ INTRODUCEŢI ESTIMAŢIILE INIŢIALE ALE PARAMETRILOR ? N

PARAMETRI INIŢIALI MA :
      0.52418

ESTIMAŢIILE DE VEROSIMILITATE MAXIMĂ ALE PARAMETRILOR
MODELULUI ARIMA (0, 1, 1) PENTRU 196 de VALORI ALE SERIEI
DUPĂ DIFERENŢIERE
    CONSTANTA DE MEDIE ALUNECĂTOARE = 0.00204
    DISPERSIA ZGOMOTULUI ALB = 0.09940

    PARAMETRI MA :
      0.69146

AB&J > ARIMA

SPECIFICAŢI STRUCTURA MODELULUI ? 1 0 1
DORIŢI SĂ INTRODUCEŢI ESTIMAŢIILE INIŢIALE ALE PARAMETRILOR ? Y
ESTIMAŢII INIŢIALE PENTRU MODELUL AR ? .2
ESTIMAŢII INIŢIALE PENTRU MODELUL MA ? .2

PARAMETRI INIŢIALI AR :
     0.20000

PARAMETRI INIŢIALI MA :
     0.20000
```

AVERTISMENT : DUPĂ 25 ITERAŢII PARAMETRII MODELULUI NU CONVERG
ESTIMAŢIILE DE VEROSIMILITATE MAXIMĂ ALE PARAMETRILOR
MODELULUI ARIMA (1, 0, 1) PENTRU 197 DE VALORI ALE SERIEI
DUPĂ DIFERENŢIERE

CONSTANTA DE MEDIE ALUNECĂTOARE = 1.69613
DISPERSIA ZGOMOTULUI ALB = 0.09741

PARAMETRI AR :
0.90060

PARAMETRI MA :
0.55382

AB&J > AGAIN

PARAMETRI INIŢIALI AR :
0.90060

PARAMETRI INIŢIALI MA :
0.55382

ESTIMAŢIILE DE VEROSIMILITATE MAXIMĂ ALE PARAMETRILOR
MODELULUI ARIMA (1, 0, 1) PENTRU 197 DE VALORI ALE SERIEI
DUPĂ DIFERENŢIERE

CONSTANTA DE MEDIE ALUNECĂTOARE = 1.65346
DISPERSIA ZGOMOTULUI ALB = 0.09731

PARAMETRI AR :
0.90311

PARAMETRI MA :
0.55899

AB&J > SMOD

NUMELE FIŞIERULUI ? MOD
MODELUL SALVAT ÎN FIŞIERUL : MOD

AB&J > ORIG
AB&J > FOREC

EPSILON ? 0.05
NUMĂR VALORI PREDICŢIE ? 8

MOMENT PREDICŢIE	PONDERI PSI	VALOARE PREDICŢIE	EROARE STD. 5.00%
1	0.34408	17.36903	0.59685
2	0.31074	17.33937	0.63119
3	0.28063	17.31258	0.65788
4	0.25343	17.28839	0.67886
5	0.22887	17.26654	0.69551
6	0.20670	17.24681	0.70880
7	0.18667	17.22900	0.71945
8	0.16858	17.21290	0.72803

AB&J > UPDATE

DATELE CITITE DIN FIŞIERUL ? BJS
197 DATE CITITE DIN FIŞIERUL : BJS

17.00000 16.60000 16.30000 16.10000 ....

NUME FIŞIER MODEL ? MOD
MODELUL ÎNCĂRCAT DIN FIŞIERUL : MOD

ESTIMAŢIILE DE VEROSIMILITATE MAXIMĂ ALE PARAMETRILOR MODELULUI ARIMA (1, 0, 1) PENTRU 197 DE VALORI ALE SERIEI DUPĂ DIFERENŢIERE
CONSTANTA DE MEDIE ALUNECĂTOARE = 1.65346
DISPERSIA ZGOMOTULUI ALB = 0.09731

PARAMETRI AR :
0.90311

PARAMETRI MA :
0.55899

VALOAREA NOII OBSREVAŢII ? 17.42

EPSILION ? 0.05
NUMĂR VALORI PREDICŢIE ? 10

MOMENT PREDICŢIE	PONDERI PSI	VALOARE PREDICŢIE	EROARE STD. 5.00 %
1	0.34408	17.35690	0.59538
2	0.31074	17.32842	0.62964
3	0.28063	17.30269	0.65626
4	0.25343	17.27946	0.67719
5	0.22887	17.25848	0.69380
6	0.20670	17.23953	0.70705
7	0.18667	17.22241	0.71768
8	0.16858	17.20696	0.72624
9	0.15224	17.19300	0.73314
10	0.13749	17.18040	0.73872

FIŞIERUL ÎN CARE SE SCRIU DATELE ? UPD
198 DATE SCRISE ÎN FIŞIERUL UPD

AB&J > EXIT

6. CONCLUZII

Pachetul AUTOB&J reprezintă un sistem de programe complet interactiv pentru modelarea şi predicţia seriilor de timp, proiectat pentru implementarea metodologiei Box-Jenkins, ce reprezintă una dintre cele mai larg utilizate şi precise tehnici de predicţie pe termen scurt.

Produsul este uşor de utilizat în practică, robust, portabil şi necesită resurse de calcul modeste.

Pachetul oferă soluţii pentru rezolvarea unor probleme de modelare şi predicţie ce apar în industrie, transporturi, agricultură, geologie, meteorologie, demografie, biologie şi medicină, ecologie, economie etc.

BIBLIOGRAFIE

Akaike, H., E. Arahata și T. Ozaki (1975). **A Computer Time Series and Control Program Package (1), (2), Computer Science Monographs, 5, 6,** The Institute of Statistical Mathematics, Tokyo.

Box, G.E.P. și G.M. Jenkins (1976). **Time Series Analysis : Forecasting and Control,** Revised Edition, Holden Day, San Francisco.

Brown, K.M. și J.E. Dennis, Jr. (1971). **On the second order convergence of Brown's derivative-free method for solving simultaneous nonlinear equations,** Yale University Department of Computer Science, Technical Report.

Dagum, E.B. (1979). **The X-11-ARIMA Seasonal Adjustment Method, Outline of the Methodology,** Statistics Canada.

Goel, P.K. și A.G. Rocco (1982). ARIMA Modelling and Forecasting : **An Interactive Program Based on IMSL Subroutine Package – II, Technical Report 82–12, Department of Statistics,** Purdue University.

Popescu, Th. D. (1983). TSPACK – **Subroutine Package for Time Series Analysis and Forecasting, Report TSA& F/1,** Institutul Central pentru Conducere și Informatică, București.

Popescu, Th. D. (1984). **Analiza și predicția seriilor de timp. Aspecte metodologice și aplicații, Raport de Cercetare TSA& F/2,** Institutul Central pentru Conducere și Informatică București.

Popescu, Th. D. (1985a). **AUTOB& J – A Computer Aided Procedure for Time Series Analysis and Forecasting, 3rd IFAC Symposium on Computer Aided Design in Control and Engineering Systems,** Technical University of Denmark, Lyngby.

Popescu, Th. D. (1985b). **Experiences with a Computer Aided Procedure for Time Series Analysis and Forecasting Using Box-Jenkins Philosophy, 2nd International Symposium on System Analysis and Simulation,** 26–31 August 1985, Berlin.

Luenberger, D.G. (1973). **Introduction to Linear and Nolinear Programming, Addison-Wesley,** Reading, Massachusetts.

Reilly, D.P. (1984). **An Overview of the PACK System and AUTOBJ/AUTOBOX,** Automatic Forecasting Systems Inc., Hatboro.

Shellswell, S.H. (1972). **A Computer Aided Procedure for Time Series Analysis and Identification of Noisy Processes (CAPTAIN), CUED/B Control/TR 25,** Cambridge.

Tertișco, M., P. Stoica și Th. Popescu (1985). **Modelarea și predicția seriilor de timp,** Editura Academiei, București.

ACTIVITATEA SECŢIEI DE AUTOMATICĂ A CNIT

În cadrul Secţiei de Automatică a CNIT a fost constituită comisia „Tehnologii, echipamente şi metode de asigurare a calităţii mijloacelor de automatizare". În cadrul acestei comisii, au fost cooptaţi specialişti din domeniile proiectării şi utilizării sistemelor de testare automată, a controlului calităţii şi fiabilităţii echipamentelor de automatizare şi pentru informatică industrială, precum şi specialişti în domeniul proiectării tehnologice şi asigurării calităţii acestor produse. Membrii comisiei provin din unităţile cele mai reprezentative din cercetare, proiectare, producţie şi din institutele de învăţămînt superior din Bucureşti, Cluj-Napoca, Timişoara, Craiova.

Conducerea comisiei este asigurată de către tov. profesor dr. ing. Marius Hăngănuţ — şef filială I.P.A. Cluj-Napoca — preşedinte, şi dr. ing. M. Vlăduţiu — Institutul Politehnic Timişoara ; dr. ing. I. Hohan — Institutul Politehnic Bucureşti vicepreşedinţi, şi ing. D. Nasui — I.I.R.U.C. Bucureşti — secretar.

Obiectivele acestei comisii constau în promovarea şi intensificarea în cadrul tuturor întreprinderilor a metodelor moderne de asigurare şi control a calităţii, fiabilităţii, mentenabilităţii şi disponibilităţii echipamentelor de automatizare, precum şi în introducerea tehnologiilor de testare automată.

Comisia îşi desfăşoară activitatea conform unui plan anual care cuprinde diverse manifestări ca : mese rotunde, dezbateri, conferinţe în întreprinderi, simpozioane.

Din planul comisiei pe anul 1985 se menţionează :

- Creşterea productivităţii şi competitivităţii producţiei în industria textilă prin automatizare ;
- Calitatea producţiei prin utilizarea aparaturii analitice automate — analizoare de gaze ;
- Tehnologii de proiectare şi testare asistată de calculator ;
- Proiectarea pentru testabilitate a echipamentelor de automatizare ;
- Metode moderne de analiză sistemică a fiabilităţii, mentenabilităţii şi disponibilităţii echipamentelor de automatizare ;
- Problematica generării automate a testelor, stadiul actual şi prespective ;
- Proiectarea şi realizarea standurilor de testare — rolul acestora în creşterea calităţii.

În atenţia comisiei stă şi activitatea de coordonare a publicării lucrărilor elaborate în domeniu. Evenimentul major al comisiei îl constituie „Simpozionul de tehnologii şi echipamente de testare automată" organizat anual la Cluj-Napoca şi care în acest an se află la a patra ediţie.

Toţi cei care sînt interesaţi în activitatea comisiei pot obţine informaţii suplimentare scriind pe adresa :

IPA, Filiala Cluj-Napoca
pt. comisia CNIT
3400 Cluj-Napoca
Str. Republicii nr. 109
Telefon 16155/951. Telex 031.391.

TEHNICI DE FACTORIZARE ÎN ALGORITMII DE CONDUCERE ADAPTIVĂ

Dr. ing. *V. Sima*
I.T.C.I.

Numeroşi algoritmi propuşi recent pentru conducerea adaptivă a sistemelor multivariabile discrete utilizează metoda recursivă a celor mai mici patrate (CMMPR). Exemple tipice sînt algoritmii CMMPR modificate sau CMMPR ponderate. Programarea directă a relaţiei de actualizare a matricei de covarianţă care apare în aceşti algoritmi implică posibile dificultăţi numerice: instabilitate, pierderea pozitivităţii. Lucrarea tratează tehnicile de factorizare aplicabile matricei de covarianţă sau inversei acesteia, care permit evitarea dificultăţilor menţionate. Sînt prezentate şi analizate noi variante ale algoritmilor de factorizare ; sînt date detalii de implementare şi recomandări pentru utilizarea unor rutine matematice performante, recent elaborate, pentru rezolvarea problemelor de calcul asociate. De asemenea, sînt discutate unele rezultate numerice obţinute şi sînt listate subprograme FORTRAN originale pentru actualizarea factorizărilor.

1. Introducere

Numeroase procese fizice necesită sisteme de reglare capabile să lucreze în diferite puncte de funcţionare. Datorită neliniarităţii, un singur model liniar nu poate descrie corespunzător un proces în toate punctele de funcţionare. În consecinţă, un sistem de reglare cu parametri fixaţi, determinaţi pe baza unui anumit model, se poate dovedi neadecvat conducerii procesului în ansamblu. Soluţii pentru problema modificării parametrilor ar fi : utilizarea mai multor legi de reglare şi a unei strategii de comutare corespunzătoare, sau folosirea unui sistem de reglare cu coeficienţi funcţionali dependenţi de parametrii modelului, precalculaţi şi memoraţi pentru fiecare punct de funcţionare. Nici una din aceste soluţii nu asigură o funcţionare optimă dacă numărul de parametri este mare, sau dacă se impun reglării cerinţe de calitate severe. În astfel de cazuri devine promiţătoare adoptarea unui sistem de reglare adaptiv,

capabil să-şi ajusteze parametrii, pe baza informaţiilor colectate din proces, pentru a compensa modificările lente dar semnificative ale caracteristicilor procesului la trecerea de la un punct de funcţionare la altul.

În ultimii ani eforturile de cercetare în domeniul sistemelor adaptive s-au intensificat considerabil. Principalele rezultate teoretice obţinute sînt stabilitatea globală a unor legi de conducere adaptivă a sistemelor monovariabile continue deterministe [1], [2] şi a unor algoritmi pentru sisteme mono- şi multivariabile discrete deterministe [3]—[6], cît şi convergenţa globală cu probabilitate 1 a unor algoritmi de conducere, predicţie şi filtrare adaptivă pentru sisteme mono- şi multivariabile discrete stocastice [7]—[14]. Unele rezultate originale au fost prezentate de autor în [15—18] şi se aplică unor algoritmi de conducere adaptivă a sistemelor de fază minimă monovariabile discrete, deterministe sau stocastice. Principalele instrumente matematice folosite în analiza legilor şi algoritmilor adaptivi sînt teoria stabilităţii Liapunov, teoria hiperstabilităţii şi, mai general, teoria pasivităţii. În cazul discret stocastic se utilizează din ce în ce mai mult funcţii Liapunov stocastice şi rezultate ale teoriei martingalelor.

Pînă în prezent s-a manifestat tendinţa de a propune diverse soluţii de conducere/predicţie adaptivă şi de a demonstra stabilitatea, respectiv convergenţa. Au existat şi unele încercări de unificare, prin adoptarea unor algoritmi suficient de generali, reprezentînd în fapt clase de algoritmi [12]. De asemenea, abordările folosite permiteau generalizări, astfel încît multe rezultate stabilite pentru anumite categorii de procese au putut fi uşor extinse la alte categorii (de exemplu, de la sisteme monovariabile la multivariabile, de la sisteme de fază minimă la cele de fază neminimă etc). Totuşi, deocamdată, nu există o teorie unitară, suficient de elaborată şi generală, ca în cazul sistemelor liniare multivariabile.

Au fost studiate cu precădere două clase de algoritmi de conducere adaptivă : algoritmi utilizînd iteraţia aproximaţiei stocastice şi algoritmi utilizînd iteraţia celor mai mici patrate (CMMPR), în diverse variante. Există o bogată experienţă numerică în simulare, privind comportarea acestor algoritmi. De asemenea, unii algoritmi au fost deja adoptaţi pentru conducerea cu calculator a unor procese tehnologice [29]. În aplicaţii, sînt preferaţi algoritmii de tip CMMPR, datorită vitezei lor de convergenţă superioare celei corespunzătoare algoritmilor de tip aproximaţie stocastică.

Implementarea algoritmilor CMMPR pe calculatoare de proces implică luarea în considerare a unor aspecte practice, cum ar fi : viteza de convergenţă, necesarul de memorie, dificultăţi numerice datorate preciziei limitate a calculelor. Ultimul aspect este cel mai important, deoarece programarea directă a algoritmilor poate conduce la calcule prost condiţionate şi la instabilitate numerică, interzicînd utilizarea on-line a codurilor respective. Adoptarea tehnicilor de factorizare a matricei de covarianţă sau a inversei acesteia (matricea de informaţie), prezentate în această lucrare, evită dificultăţile numerice asociate implementării directe a algoritmilor. Astfel de tehnici au fost folosite în domeniul estimării secvenţiale a parametrilor [19—23] şi în problemele de optimizare neliniară utilizînd metode cvasi-Newton [24—26].

Orice algoritm de conducere adaptivă parcurge la fiecare tact două etape de calcul: actualizarea estimației parametrilor $\theta(t)$ și determinarea mărimii de comandă $u(t)$, prelucrînd estimația $\theta(t)$ și informația curentă disponibilă în proces $\Phi(t)$. Lucrarea tratează etapa fundamentală de actualizare a estimației, utilizînd tehnici de factorizare fie a matricei de covarianță, fie a matricei de informație. Sînt discutate în detaliu, din punctul de vedere al implementării, atît factorizarea de tip rădăcină patrată [19], [22], cît și factorizarea de tip U—D [20, 21]. Principalul instrument matematic adoptat pentru actualizarea factorizării îl constituie rotațiile plane, standard [27], sau modificate [28]. Acestea oferă baza pentru o tratare unitară a diferiților algoritmi de factorizare și asigură în final proprietățile numerice bune ale acestora. Sînt obținute pe o nouă cale unele rezultate cunoscute; alte rezultate sînt originale. Sînt discutate detalii de implementare și se fac recomandări pentru utilizarea unor rutine matematice performante, recent elaborate, pentru rezolvarea problemelor implicate.

Lucrarea este organizată astfel. În secțiunea 2 sînt prezentați cîțiva algoritmi de conducere adaptivă tipici. În secțiunea 3 se tratează problema actualizării estimației și se obțin relațiile de bază care vor fi folosite în continuare. În secțiunile 4 și 5 se prezintă tehnicile de factorizare a matricei de informație și, respectiv, de covarianță. În secțiunea 6 se discută unele rezultate numerice obținute. Secțiunea 7 conține concluziile lucrării. În Anexa 1 se deduce un algoritm de actualizare a factorizării unei matrici simetrice pozitiv definite după adunarea (scăderea) unei diade simetrice. În Anexele 2 și 3 sînt listate coduri tipice, elaborate de autor, pentru actualizarea factorizărilor.

2. Algoritmi tipici pentru conducere adaptivă

Înainte de a prezenta cîțiva algoritmi de conducere adaptivă vom indica temeiurile care stau la baza formulării lor.

Considerăm un sistem multivariabil descris de următorul model intrare/ieșire (de tip ARMA)*

$$A(q^{-1})\,y(t)=q^{-d}\,[B(q^{-1})]\,u(t)+[C(q^{-1})]\,w(t) \tag{1}$$

unde $y(t)\in\mathbb{R}^m$ este mărimea de ieșire a procesului care trebuie condus, $u(t)\in\mathbb{R}^r$ este mărimea de intrare (comanda), $w(t)\in\mathbb{R}^m$ este o mărime perturbatoare, q^{-d} reprezintă o întîrziere pură de d tacte, $A(q^{-1})$ este un polinom scalar în operatorul de întîrziere cu un tact q^{-1}, iar $[B(q^{-1})]$ și $[C(q^{-1})]$ sînt matrice ale căror elemente sînt polinoamele scalare $B_{ij}(q^{-1})$ și $C_{ij}(q^{-1})$, respectiv:

$$A(q^{-1})=1+a_1q^{-1}+\ldots+a_nq^{-n} \tag{2.a}$$

$$B_{ij}(q^{-1})=(B_{ij})_0+(B_{ij})_1\,q^{-1}+\ldots+(B_{ij})_s\,q^{-s} \tag{2.b}$$

$$C_{ij}(q^{-1})=(C_{ij})_0+(C_{ij})_1\,q^{-1}+\ldots+(C_{ij})_r\,q^{-r},\quad (C_{ij})_0=\begin{cases}1, & i=j\\ 0, & i\neq j\end{cases} \tag{2.c}$$

* Rezultatele pot fi obținute și pentru anumite modele de stare [13, 14].

Coeficienţii polinoamelor sînt necunoscuţi şi numai semnalele $\{y(t)\}$ şi $\{u(t)\}$ sînt disponibile. Se presupune că procesul îndeplineşte anumite condiţii de stabilitate inversă şi pasivitate [13, 14] cerute pentru convergenţa algoritmilor de conducere adaptivă. De asemenea, şirul $\{w(t)\}$ este un proces stocastic (de tip zgomot alb) cu medie nulă, satisfăcînd anumite condiţii de independenţă condiţională şi de mărginire.

Obiectivul conducerii constă în determinarea comenzii $\{u(t)\}$, utilizînd o lege de reglare cu reacţie, care să stabilizeze sistemul şi să asigure că $\{y(t)\}$ urmăreşte cu dispersie condiţională minimă o ieşire dorită, mărginită, $\{y^*(t)\}$.

Modelul (1) se poate rescrie în forma de predicţie liniară [9], [13]

$$y(t)=\widehat{y^*}(t)+v(t) \tag{3}$$

unde

$$\widehat{y^*}(t)=q^{-d}\{[G(q^{-1})]\widehat{y^*}(t)+[\alpha(q^{-1})]y(t)+[\beta(q^{-1})]u(t)\}\triangleq\theta^T x(t-d) \tag{4.a}$$

$$x(t)^T\triangleq[y(t)^T,\ldots,y(t-n_1)^T,u(t)^T,\ldots,u(t-n_2)^T,\widehat{y^*}(t)^T,\ldots,$$
$$\ldots,\widehat{y^*}(t-n_3)^T] \tag{4.b}$$

$$v(t)\triangleq[F(q^{-1})]w(t)=\sum_{i=0}^{d-1}F_i w(t-i),\ F_i\in\mathbb{R}^{m\times m},\ F_0=I_m \tag{4.c}$$

În relaţiile (4), indicele superior T denotă transpunerea, I_m reprezintă matricea unitate de ordin m, $\widehat{y^*}(t+d)$ este predicţia liniară optimală cu d paşi înainte a lui $y(t)$, determinată utilizînd informaţiile pînă la momentul t. Matricele polinomiale $[G(q^{-1})]$, $[\alpha(q^{-1})]$, $[\beta(q^{-1})]$ şi $[F(q^{-1})]$ depind de coeficienţii polinoamelor $A(.)$, $B_{ij}(.)$ şi $C_{ij}(.)$; n_1, n_2 şi n_3 sînt gradele maxime (finite) ale polinoamelor din $[G]$, $[\alpha]$ şi respectiv $[\beta]$; $\theta\in\mathbb{R}^{p\times m}$ este matricea parametrilor definită în (4.a), iar $p=m(n_1+n_3+2)+r(n_2+1)$.

În ipoteza parametrilor cunoscuţi (deci θ cunoscut), obiectivul conducerii adaptive este atins dacă se alege $u(t)$ ca soluţie a sistemului liniar implicit definit

$$\theta^T x(t)=y^*(t+d) \tag{5}$$

În cazul parametrilor necunoscuţi, matricea θ se înlocuieşte cu o estimaţie a sa $\theta(t)$, actualizată de un algoritm recursiv, iar $\widehat{y^*}(t)$ se înlocuieşte cu o mărime calculabilă, funcţie de informaţia disponibilă $\Phi(t)$.

Din formularea generală de mai sus se pot obţine uşor prin particularizare formulările valabile în cazul sistemelor monovariabile sau deterministe

$$(w(t)=0,\ v(t)=0).$$

În continuare, vom considera cîţiva algoritmi tipici de conducere adaptivă CMMPR.

Algoritmul CMMPR1, pentru sisteme monovariabile deterministe [15, 16], [18]:

$$\theta(t)=\theta(t-1)+\frac{a(t)P(t-2)\Phi(t-d)}{\lambda(t)+a(t)\Phi(t-d)^T P(t-2)\Phi(t-d)}[y(t)-\theta(t-1)^T\Phi(t-d)] \tag{6.a}$$

$$P(t-1)=\left[P(t-2)-\frac{a(t)P(t-2)\Phi(t-d)\Phi(t-d)^T P(t-2)}{\lambda(t)+a(t)\Phi(t-d)^T P(t-2)\Phi(t-d)}\right]/\lambda(t) \tag{6.b}$$

$$\theta(t)^T\Phi(t)=y^*(t+d) \tag{6.c}$$

$$\Phi(t)^T=[y(t),\ldots,y(t-n+1),\ u(t),\ldots,u(t-m-d+1)] \tag{6.d}$$

unde $P(.)\in\mathbb{R}^{p\times p}$ este matricea de covarianţă, $\lambda(t)\in(0,1]$ este un factor de ponderare permiţînd eliminarea („uitarea") informaţiilor vechi, iar $a(t)$ este un număr pozitiv ales astfel încît să se evite împărţirea la zero în (6.c) la determinarea lui $u(t)$. Alegeri posibile pentru $\lambda(t)$, asigurînd mărginirea urmei matricei $P(t)^{-1}$ sînt indicate în [6], [16]. Un algoritm similar, pentru cazul $d=t$, se găseşte în [3].

Algoritmul multirecursiv CMMPR2; CMMPR modificate pentru sisteme monovariabile stocastice [15], [17, 18]:

$$\theta(t)=\theta(t-d)+\frac{P(t-2d)\Phi(t-d)}{1+\Phi(t-d)^T P(t-2d)\Phi(t-d)}[y(t)-\theta(t-d)^T\Phi(t-d)] \tag{7.a}$$

$$P'(t-d)=P(t-2d)-\frac{P(t-2d)\Phi(t-d)\Phi(t-d)^T P(t-2d)}{1+\Phi(t-d)^T P(t-2d)\Phi(t-d)},$$

$$P(-d)=\ldots=P(-1)=\varepsilon I_p,\ \varepsilon \text{ mic} \tag{7.b}$$

$$r(t-d)=r(t-2d)[1+\Phi(t-d)^T P(t-2d)\Phi(t-d)],$$

$$r(-d)=\ldots=r(-1)=\rho>0 \tag{7.c}$$

$$P(t-d)=P'(t-d),\ \text{dacă } r(t-d)\,\mathrm{tr}[P'(t-d)]<K,\quad K>0 \tag{7.d}$$

$$P(t-d)=\frac{K}{r(t-d)\,\mathrm{tr}[P'(t-d)]}P'(t-d),\ \text{astfel} \tag{7.e}$$

$$\theta(t)^T\Phi(t)=y^*(t+d) \tag{7.f}$$

$$\Phi(t)^T=[y(t),\ldots,y(t-n_1),\ u(t),\ldots,u(t-n_2),\ \bar{y}(t),\ldots,\bar{y}(t-n_3)] \tag{7.g}$$

$$\bar{y}(t)=\theta(t)^T\Phi(t-d),\ \bar{y}(\tau)=0,\ \tau\leqslant d-1. \tag{7.h}$$

Mai sus, $\mathrm{tr}[A]$ denotă urma matricei A. Algoritmul generalizează pe acela propus în [11] pentru $d=1$.

Algoritmul CMMPR3; CMMPR ponderate pentru sisteme multivariabile stocastice [13—15]:

$$\theta(t)=\theta(t-1)+\frac{a(t)P(t-2)\Phi(t-d)}{1+a(t)\Phi(t-d)^T P(t-2)\Phi(t-d)}[y(t)-\theta(t-1)^T\Phi(t-d)]^T \tag{8.a}$$

$$P(t-1)=P(t-2)-\frac{a(t)P(t-2)\Phi(t-d)\Phi(t-d)^T P(t-2)}{1+a(t)\Phi(t-d)^T P(t-2)\Phi(t-d)},\quad P(-1)=\varepsilon I_p \tag{8.b}$$

$$\theta(t)^T\Phi(t)=y^*(t+d) \tag{8.c}$$

$$\Phi(t)^T=[y(t)^T,\ldots,y(t-n_1)^T,\ u(t)^T,\ldots,u(t-n_2)^T,\ \bar{y}(t)^T,\ldots,$$
$$\ldots,\bar{y}(t-n_3)^T] \tag{8.d}$$

$$\bar{y}(t)=\theta(t)^T\Phi(t-d) \tag{8.e}$$

iar a (t) este un coeficient de ponderare ales în funcţie de măsura de stabilitate M (t), definită de

$$M(t)=\Phi(t-d)^T P(t-2)\Phi(t-d), \tag{9}$$

astfel încît să se asigure convergenţa algoritmului.

Menţionăm că pentru algoritmii prezentaţi succint mai sus există demonstraţii matematice de stabilitate, respectiv convergenţă cu probabilitate [1]. Algoritmi similari sînt folosiţi şi la regulatoarele autoacordabile [29, 30].

3. Actualizarea estimaţiei parametrilor

Etapa de calcul esenţială a algoritmilor de conducere adaptivă o constituie actualizarea estimaţiei parametrilor. Examinînd algoritmii prezentaţi în secţiunea precedentă, rezultă că ecuaţiile recursive de bază sînt următoarele :

$$\theta(t)=\theta(t-1)+\frac{a(t)P(t-2)\Phi(t-d)}{\lambda(t)+a(t)\Phi(t-d)^T P(t-2)\Phi(t-d)}[y(t)-\theta(t-1)^T\Phi(t-d)]^T \tag{10}$$

$$P(t-1)=\left[P(t-2)-\frac{a(t)P(t-2)\Phi(t-d)\Phi(t-d)^T P(t-2)}{\lambda(t)+a(t)\Phi(t-d)^T P(t-2)\Phi(t-d)}\right]/\lambda(t) \tag{11}$$

unde $\theta(.)\in\mathbb{R}^{p\times m}$, $P(.)\in\mathbb{R}^{p\times p}$, $\Phi(.)\in\mathbb{R}^p$, $y(.)\in\mathbb{R}^m$, $a(.)$, $\lambda(t)\in\mathbb{R}$ au fost definite anterior. Ecuaţiile de mai sus sînt iniţializate cu $\theta_0=\theta(0)$ şi $P_{-1}=P(-1)>0$. De notat că şi în cazul algoritmului CMMPR2, putem considera că intervin relaţiile (10) şi (11).

Menţionăm că ecuaţii similare cu (10) şi (11) apar în problemele de estimare secvenţială discretă [21, 22]. Din aplicaţiile teoriei estimării parametrice, este bine cunoscut faptul că implementarea directă pe calculatoare a recurenţei (11), utilizînd de exemplu produse exterioare, este numeric instabilă. Pot

apare covarianţe negative [23] şi chiar matrice de „covarianţă" negativ definite. Acest lucru se poate obţine şi pentru exemple de mici dimensiuni, cum este cel dat în continuare [20], [15]:

Exemplul 1: Fie $a(t)\equiv 1$, $\lambda(t)\equiv 1$, $d=1$ şi $\Phi(t-1)^T=[1\ \varepsilon]$, $\Phi(t)^T=[1,\ 1]$, $P(t-2)=\sigma^2 I_2$, $\sigma=1/\varepsilon$, unde ε este ales astfel încît $0<\varepsilon<<1$ şi $fl(1+\varepsilon^2)=1$. Notaţia $fl(x)$ denotă rezultatul obţinut evaluînd expresia x pe un calculator cu aritmetică în virgulă mobilă. Pentru un calculator cu precizia relativă de 6 cifre zecimale exacte, putem presupune $\varepsilon=10^{-4}$. Este uşor de verificat (vezi [15]) că dacă $P(.)$ se actualizează direct utilizînd relaţiile (corespunzătoare filtrului Kalman clasic)

$$K(t-1)=P(t-2)\Phi(t-d),\ b(t)=a(t)/[\lambda(t)+a(t)\Phi(t-d)^T K(t-1)] \tag{12. a}$$

$$P(t-1)=[P(t-2)-b(t)K(t-1)K(t-1)^T]/\lambda(t) \tag{12. b}$$

se obţine

$$fl[P(t)]=\frac{1}{1-2\varepsilon}\begin{bmatrix}-1 & 1\\ 1 & 0\end{bmatrix}.$$

Aşadar, la a doua iteraţie matricea $P(\cdot)$ calculată devine negativ definită.

Problemele numerice sînt probabil mai importante pentru implementarea pe calculator a algoritmilor de conducere adaptivă. Aceşti algoritmi trebuie să funcţioneze on-line pe perioade de timp practic infinite. Mai mult, pentru conducerea adaptivă în aplicaţiile industriale se utilizează frecvent microcalculatoare cu lungime de cuvînt mică, deci cu precizie de reprezentare a numerelor reale redusă.

Soluţii numerice bune se obţin prin factorizarea matricei P, ori a inversei acesteia şi prin actualizarea factorilor respectivi. În acest scop, au fost propuse tehnici de factorizare de tip rădăcină patrată (Cholesky) [19], sau de tip U—D [20], [23].

În continuare, vom deduce cîteva relaţii utilizate în secţiunile viitoare. Pentru a simplifica notaţia, argumentele vor fi omise de aici încolo, iar cantităţile actualizate vor fi marcate prin simbolul „$\wedge$". Relaţiile (10) şi (11) se rescriu astfel

$$\hat{\theta}=\theta+\frac{aP\Phi}{\lambda+a\Phi^T P\Phi}(y-\theta^T\Phi)^T \tag{13}$$

$$\hat{P}=\left(P-\frac{aP\Phi\Phi^T P}{\lambda+a\Phi^T P\Phi}\right)/\lambda. \tag{14}$$

Utilizînd (14), din (13) rezultă

$$\hat{\theta}=\left(I_p-\frac{aP\Phi\Phi^T}{\lambda+a\Phi^T P\Phi}\right)\theta+\frac{aP\Phi}{\lambda+a\Phi^T P\Phi}y^T=\lambda\hat{P}P^{-1}\theta+a\hat{P}\Phi y^T. \tag{15}$$

Notînd $Q=P^{-1}$, matricea de informaţie, (15) şi (14) implică, respectiv

$$\hat{Q}\hat{\theta}=\lambda Q\theta+a\Phi y^T \tag{16}$$

$$\hat{Q}=\lambda Q+a\Phi\Phi^T. \tag{17}$$

Relaţiile (16) şi (17) sînt relaţiile de bază care vor fi utilizate în secţiunea următoare. Relaţii similare cu (17) apar de asemenea în metodele cvasi-Newton din problemele de optimizare neliniară [24—26].

4. Tehnici de factorizare a matricei de informaţie

Vom prezenta mai întîi factorizarea de tip rădăcină pătrată (Cholesky) a matricei de informaţie şi apoi factorizarea de tip „U—D".

Factorizarea tip rădăcină patrată

Considerăm factorizarea $Q=R^TR$, unde R este o matrice superior triunghiulară, şi notăm

$$z=R\theta. \tag{18}$$

Fie G o matrice ortogonală de ordin $p+1$ (adică $GG^T=G^TG=I_{p+1}$), astfel ca

$$G\begin{bmatrix}\lambda^{1/2}R\\ a^{1/2}\Phi^T\end{bmatrix}=\begin{bmatrix}\widehat{R}\\ 0\end{bmatrix} \tag{19}$$

unde $\widehat{R}$ este o matrice superior triunghiulară. Observăm că $\widehat{R}$ este un factor al lui $\widehat{Q}$, deoarece cu (17)

$$\widehat{R}^T\widehat{R}=[\lambda^{1/2}R^T \quad a^{1/2}\Phi]\begin{bmatrix}\lambda^{1/2}R\\ a^{1/2}\Phi^T\end{bmatrix}=\lambda R^TR+a\Phi\Phi^T=\widehat{Q}.$$

Ecuaţiile (16) şi (17) se pot rescrie astfel

$$\widehat{R}^T\widehat{z}=\lambda R^Tz+a\Phi y^T \tag{20}$$

$$\widehat{R}^T\widehat{R}=\lambda R^TR+a\Phi\Phi^T. \tag{21}$$

Notînd

$$\begin{bmatrix}\bar{z}\\ \widehat{\rho}^T\end{bmatrix}=G\begin{bmatrix}\lambda^{1/2}z\\ a^{1/2}y^T\end{bmatrix} \tag{22}$$

din (19)—(22) avem,

$$\widehat{R}^Tz=[\lambda^{1/2}R^T \quad a^{1/2}\Phi]\begin{bmatrix}\lambda^{1/2}z\\ a^{1/2}y^T\end{bmatrix}=\begin{bmatrix}\lambda^{1/2}R\\ a^{1/2}\Phi^T\end{bmatrix}^T G^TG\begin{bmatrix}\lambda^{1/2}z\\ a^{1/2}y^T\end{bmatrix}=\begin{bmatrix}\widehat{R}\\ 0\end{bmatrix}^T\begin{bmatrix}\bar{z}\\ \widehat{\rho}^T\end{bmatrix}=\widehat{R}^T\bar{z}.$$

deci $\widehat{z}=\bar{z}$, deoarece $\widehat{R}$ este nesingulară.

Astfel, procedura pentru actualizarea lui θ cuprinde următoarele etape :

1) Determină G şi $\hat{R}$ astfel încît (19) este satisfăcută.
2) Calculează $\hat{z}$ aplicînd G asupra matricei $[\lambda^{1/2}z^T \; a^{1/2}y]^T$.
3) Rezolvă sistemele $\hat{R}\hat{\theta}=\hat{z}$.

Matricea ortogonală G poate fi determinată ca produs a p rotaţii plane [27], $G=G_pG_{p-1}\ldots G_1$, unde G_i este o rotaţie în planul $(i, p+1)$ dată de

$$G_i=\begin{array}{c}\begin{matrix} & \downarrow i & & \downarrow p+1 \end{matrix}\\ \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & c_i & 0 & s_i \\ 0 & 0 & I & 0 \\ 0 & -s_i & 0 & c_i \end{bmatrix}\end{array}\begin{matrix} \\ \\ \leftarrow i \\ \\ \leftarrow p+1 \end{matrix}, \quad c_i^2+s_i^2=1. \tag{23}$$

Pentru $\lambda=a=1$, procedura de mai sus se reduce la cea descrisă în documentaţia pachetului LINPACK [27]. Subrutina SCHUD din acest pachet implementează primii doi paşi, iar subrutina STRSL poate fi folosită pentru a rezolva sistemele liniare de la pasul 3. SCHUD poate fi direct utilizată şi în cazul $\lambda \neq 1$, $a \neq 1$ dacă, înainte de a o apela, matricele R şi z sînt multiplicate prin $\lambda^{1/2}$, şi vectorii Φ şi y sînt multiplicaţi prin $a^{1/2}$. Totuşi, aceste operaţii de preprelucrare implică $^1/_2p^2+(m+1{,}5)\,p+m$ multiplicări suplimentare. Efortul de calcul poate fi redus, prin modificarea şi rafinarea procedurii precedente, după cum se indică în algoritmul următor.

Algoritmul 1: Actualizează θ utilizînd factorizarea tip rădăcină patrată a lui P^{-1}.

Intrare : R, z ($R^TR=P^{-1}$, $z=R\theta$).

Ieşire : $\hat{R}$, $\hat{z}$, $\hat{\theta}$ (suprascriind R, z, şi θ, respectiv).

1) $\Phi := a^{1/2}\Phi$; $y := a^{1/2}y$.
2) Pentru $i=1, 2, \ldots, p$

   1) $r_{ii} := \lambda^{1/2}r_{ii}$.

   2) Determină $g_i=\begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix}$, astfel încît $\begin{bmatrix} r_{ii} \\ 0 \end{bmatrix} := g_i\begin{bmatrix} r_{ii} \\ \Phi_i \end{bmatrix}$.

   3) Pentru $i < p$

$$\begin{bmatrix} r_{i,\,i+1} \ldots r_{ip} \\ \Phi_{i+1} \ldots \Phi_p \end{bmatrix} := \begin{bmatrix} \lambda^{1/2}c_i & s_i \\ -\lambda^{1/2}s_i & c_i \end{bmatrix}\begin{bmatrix} r_{i,\,i+1} \ldots r_{ip} \\ \Phi_{i+1} \ldots \Phi_{ip} \end{bmatrix}.$$

   4)

$$\begin{bmatrix} z_{i1} \ldots\ldots z_{im} \\ y_1 \ldots\ldots y_m \end{bmatrix} := \begin{bmatrix} \lambda^{1/2}c_i & s_i \\ -\lambda^{1/2}s_i & c_i \end{bmatrix}\begin{bmatrix} z_{i1} \ldots\ldots z_{im} \\ y_1 \ldots\ldots y_m \end{bmatrix}.$$

3) $\theta = R^{-1}z$.

Observații

1) La sfîrșitul algoritmului 1, R, z, și θ conțin cantitățile actualizate $\hat{R}$, $\hat{z}$, și respectiv $\hat{\theta}$. Rotațiile g_i sînt determinate utilizînd matricele $[\lambda^{1/2}R^T \;\; a^{1/2}\Phi]^T$. La pașii 2.3 și 2.4 se aplică

$$g_i^\lambda = \begin{bmatrix} \lambda^{1/2}c_i & s_i \\ -\lambda^{1/2}s_i & c_i \end{bmatrix};$$

rezultatul este identic cu cel care ar fi obținut utilizînd direct SCHUD pentru $\lambda^{1/2}R$, $a^{1/2}\Phi^T$, $\lambda^{1/2}z$, și $a^{1/2}y^T$. Soluția propusă aici necesită doar 4p+m multiplicări suplimentare pentru $\lambda \neq 1$ și $a \neq 1$. Pentru a calcula rotațiile plane g_i^λ se poate utiliza subrutina SROTG din pachetul BLAS [28]. Subrutina SROT, de asemenea din BLAS, trebuie modificată pentru a aplica g_i^λ.

2) Costul total al algoritmului este de $(^1/_2 m+2)p^2+(4{,}5m+7)p+m$ multiplicări și p+2 rădăcini patrate. În cazul monovariabil numărul este $2{,}5p^2+11{,}5p$.

3) Algoritmul aplică rotațiile pe linii. Pentru a opera pe coloane, se pot adopta factorizările $Q=UU^T$, sau $Q=LL^T$, unde U și L sînt matrice superior și, respectiv, inferior triunghiulare. Alternativ, se poate recurge la tehnica folosită în LINPACK, dar acest lucru ar necesita 4p locații suplimentare de memorie pentru a memora c_i, s_i, $\lambda^{1/2}c_i$, și $\lambda^{1/2}s_i$, pentru $i=1, \ldots, p$.

4) Măsura de stabilitate M (t), definită de relația (9) și utilizată în algoritmul CMMPR3 poate fi calculată eficient astfel $M=\Phi^T Q^{-1}\Phi=\|R^{-T}\Phi\|^2$. Deci, pentru calculul ei sînt necesare numai $^1/_2 p^2+1{,}5p$ multiplicări.

În Anexa 2 este listată subrutina SQRINF care actualizează factorizarea UU^T a matricei de informație, pentru sisteme monovariabile, folosind tehnica prezentată mai sus. SQRINF apelează subrutina SROTG din BLAS pentru determinarea rotațiilor.

Factorizarea LDL^T

Luînd în considerare observația 3, să considerăm acum factorizarea $Q=LDL^T$, unde L este o matrice inferior triunghiulară, iar D este o matrice diagonală. Fie G o matrice ortogonală, astfel încît

$$G\begin{bmatrix} \lambda^{1/2}D^{1/2}L^T \\ a^{1/2}\Phi^T \end{bmatrix} = \begin{bmatrix} \hat{D}^{1/2}\hat{L}^T \\ 0 \end{bmatrix} \tag{24}$$

unde $\hat{D}$ este diagonală și $\hat{L}$ este inferior triunghiulară. Din (24) și (17) se obține ușor că $\hat{Q}=\hat{L}\hat{D}\hat{L}^T$. Definind $z=L^T\theta$ și

$$\begin{bmatrix} \hat{D}^{1/2}\bar{z} \\ \hat{\rho}^T \end{bmatrix} = G\begin{bmatrix} \lambda^{1/2}D^{1/2}z \\ a^{1/2}y^T \end{bmatrix} \tag{25}$$

din (16), (24) și (25) avem

$$\hat{L}\hat{D}\hat{z}=\lambda LDz+a\Phi y^T=[\lambda^{1/2}LD^{1/2} \;\; a^{1/2}\Phi]\, G^TG\begin{bmatrix} \lambda^{1/2}D^{1/2}z \\ a^{1/2}y^T \end{bmatrix}=\hat{L}\hat{D}\bar{z} \tag{26}$$

deci $\hat{z}=\bar{z}$. Astfel, procedura de actualizare a lui θ cuprinde următorii paşi:

1) Determină G, $\hat{D}$ şi $\hat{L}$, astfel încît să fie satisfăcută relaţia (24).

2) Calculează $\hat{z}$ aplicînd G asupra matricei $[\lambda^{1/2}z^TD^{1/2} \quad a^{1/2}y]^T$.

3) Rezolvă $\hat{L}^T\hat{\theta}=\hat{z}$.

Matricea G poate fi obţinută ca produs a p rotaţii plane, $G=G_pG_{p-1}\ldots G_1$. Vom utiliza rotaţii plane modificate, evitînd astfel rădăcinile patrate. Fie c_j şi s_j cosinusul şi, respectiv, sinusul care determină rotaţia G_j. Atunci putem scrie (vezi Anexa 1 pentru detalii)

$$\begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix}\begin{bmatrix} \lambda^{1/2}d_j^{1/2} & 0 \\ 0 & (a^j)^{1/2} \end{bmatrix}=\begin{bmatrix} \hat{d}_j^{1/2} & 0 \\ 0 & (a^{j+1})^{1/2} \end{bmatrix}H_j \tag{27}$$

unde $a^1=a$, matricea H_j, de ordin 2, are în general două elemente egale cu 1, iar indicele superior j denotă valorile variabilelor la a j—a etapă de calcul. Se poate utiliza subrutina SROTMG din pachetul BLAS [28], pentru a calcula H_j, $\hat{d}_j$ şi a^{j+1}. La pasul 2 al procedurii avem

$$\begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix}\begin{bmatrix} \lambda^{1/2}d_j^{1/2} & 0 \\ 0 & (a^j)^{1/2} \end{bmatrix}\begin{bmatrix} z_{j1} & \ldots & z_{jm} \\ y_1^j & \ldots & y_m^j \end{bmatrix}=\begin{bmatrix} \hat{d}_j^{1/2} & 0 \\ 0 & (a^{j+1})^{1/2} \end{bmatrix}H_j\begin{bmatrix} z_{j1} & \ldots & z_{jm} \\ y_1^j & \ldots & y_m^j \end{bmatrix}$$

aşa încît în final obţinem

$$\hat{z}=H'_p\ldots H'_1\begin{bmatrix} z \\ y^T \end{bmatrix},$$

unde

$$H'_j=\begin{array}{cc} \begin{array}{cccc} & \downarrow j & & \downarrow p+1 \end{array} & \\ \begin{bmatrix} I & 0 & 0 & 0 \\ 0 & h_{11} & 0 & h_{12} \\ 0 & 0 & I & 0 \\ 0 & h_{21} & 0 & h_{22} \end{bmatrix} & \begin{matrix} \\ \leftarrow j \\ \\ \leftarrow p+1 \end{matrix} \end{array}$$

şi $H_j=(h_{ik})$. Astfel, rezultă următorul algoritm:

Algoritmul 2; Actualizează θ utilizînd factorizarea LDL^T a lui P^{-1}.

Intrare: D, L, z ($LDL^T=P^{-1}$, $z=L^T\theta$).

Ieşire: $\hat{D}$, $\hat{L}$, $\hat{z}$, $\hat{\theta}$ (suprascriind D, L, z, şi respectiv θ).

1) Pentru $j=1, 2, \ldots, p$

   1) $d_j := \lambda d_j$.

   2) Utilizează SROTMG cu intrările d_j, a, l_{jj}, şi Φ_j pentru a determina H_j, şi pentru a actualiza d_j, a, şi l_{jj}.

3) Pentru $j < p$

$$\begin{bmatrix} l_{j+1,j} \dots l_{pj} \\ \Phi_{j+1} \dots \Phi_p \end{bmatrix} := H_j \begin{bmatrix} l_{j+1,j} \dots l_{pj} \\ \Phi_{j+1} \dots \Phi_p \end{bmatrix}.$$

4)

$$\begin{bmatrix} z_{j1} \dots\dots z_{jm} \\ y_1 \dots\dots y_m \end{bmatrix} := H_j \begin{bmatrix} z_{j1} \dots\dots z_{jm} \\ y_1 \dots\dots y_m \end{bmatrix}.$$

2) $\theta = L^{-T}z$.

Observații

5) La sfîrșitul algoritmului 2, D, L, z, și θ conțin cantitățile actualizate $\hat{D}$, $\hat{L}$, $\hat{z}$, și respectiv, $\hat{\theta}$. Pentru a aplica H_j la pașii 1.3 și 1.4 se poate utiliza subrutina SROTM din pachetul BLAS.

6) Algoritmul necesită în general $(^1/_2 m+1)\,p^2+(2{,}5m+10)\,p$ multiplicări. Numărul de operații poate fi de două ori mai mare în cel mai nefavorabil caz, adică atunci cînd matricele H_j nu au nici un element egal cu 1, pentru orice j. Acest caz este extrem de rar (vezi Anexa 1).

7) Algoritmul aplică rotațiile plane modificate, pe coloane asupra lui L și pe linii asupra lui z.

8) Măsura de stabilitate $M = \Phi^T P \Phi$ poate fi calculată simplu ca $\| L^{-T} \|^2_{D^{-1}}$.

5. Tehnici de factorizare a matricei de covarianță

Vom discuta aici mai în detaliu tehnica de factorizare U—D, deoarece tehnica rădăcinii pătrate este bine cunoscută [19], [31, 32]. În plus, algoritmul de tip rădăcină pătrată se poate obține prin particularizarea algoritmului de actualizare a factorizării U—D.

Tehnica de factorizare U—D

Considerăm factorizarea $P = UDU^T$, unde U este o matrice superior triunghiulară, iar D este matrice diagonală, și notăm

$$f = U^T\Phi,\quad v = Df,\quad K = Uv,\quad b = a/(\lambda + a\Phi^T P \Phi). \tag{28}$$

Ecuația (14) poate fi rescrisă [20], [23]

$$\hat{P} = \hat{U}\hat{D}\hat{U}^T = U\,[D - b\,(DU^T\Phi)\,(DU^T\Phi)^T]\,U^T/\lambda = U\,(D - bvv^T)\,U^T/\lambda. \tag{29}$$

Fie $\bar{U}$ și $\bar{D}$ factorii U—D ai matricei $D - bvv^T$, adică $\bar{U}\bar{D}\bar{U}^T = D - bvv^T$. Atunci, din (29) putem identifica

$$\hat{U} = U\bar{U},\quad \hat{D} = \bar{D}/\lambda$$

deoarece U și $\bar{U}$ sînt matrice superior triunghiulare. Deci, actualizarea factorizării U—D a lui P este reductibilă la un caz special de actualizare a factorizării

unei matrice pozitiv definite după o modificare „de rang 1" (adică, după adăugarea, respectiv scăderea, unei diade simetrice), discutată în Anexa 1. Nu vom utiliza aici rotaţii plane modificate, deoarece prezenţa semnului „—" poate produce dificultăţi numerice, după cum se va arăta la exemplul 2 de mai jos. Factorii $\bar{U}$ şi $\bar{D}$ pot fi determinaţi particularizînd algoritmul din Anexa II din [23], ale cărui relaţii de bază sînt regăsite pe o cale nouă în Anexa 1. Astfel, din (A9) obţinem, pentru $j=p, p-1, \ldots, 1$

$$\bar{d}_j=d_j-b_j(v_j)^2,\ b^{j-1}=b^j d_j/\bar{d}_j,\ \bar{u}_{ij}=-b_j v_j v_i/\bar{d}_j\ (i<j),\ \bar{u}_{jj}=1 \tag{30}$$

şi $v_i^{j-1}=v_i^j=v_i$, $i \leqslant j$, $b^p=b$, deoarece $D-bvv^T=I_pDI_p^T-bvv^T$. Din nefericire, algoritmul bazat pe aceste relaţii este însoţit de dificultăţi numerice. Din primele două egalităţi (30) obţinem

$$1/b^{j-1}=\bar{d}_j/(b^j d_j)=1/b^j-v_j^2/d_j$$

astfel încît, notînd $a^j=1/b^j$, rezultă următoarele relaţii recursive stabile

$$a^j=a^{j-1}+v_j^2/d_j,\ \bar{d}_j=d_j a^{j-1}/a^j,\ \bar{u}_{ij}=-v_j v_i/(a^{j-1}d_j) \tag{31}$$

şi putem alege $a^0=1/b-\sum_{j=1}^{p} v_j^2/d_j$, astfel încît $a^p=1/b$. A fost astfel regăsită soluţia propusă în [26].

Ţinînd acum seama de structura matricei $\bar{U}$, produsul $U\bar{U}$ poate fi evaluat în numai o (p^2) multiplicări. Într-adevăr, să observăm că $\bar{U}$ rezultă în forma $\bar{U}=[\bar{U}_1\bar{U}_2\ldots\bar{U}_p]$, unde $\bar{U}_j=[s^j v_1 s^j v_2 \ldots s^j v_{j-1}\ 1\ 0\ldots 0]^T$ şi $s^j=-v_j/(a^{j-1}d_j)$. Atunci,

$$\hat{U}_j=U\bar{U}_j=s^jU\,[v_1v_2\ldots v_{j-1}\ 1/s^j\ 0\ldots 0]^T=U_j+s^jK_{j-1} \tag{32}$$

unde am notat $K_{j-1}=\sum_{i=1}^{j-1} v_iU_i$.

Astfel, se obţine următorul algoritm care constituie o generalizare a algoritmului propus în [23].

Algoritmul 3 : Actualizează θ utilizînd factorizarea UDU^T a lui P.

Intrare : $D, U, \theta\ (UDU^T=P)$.

Ieşire : $\hat{D}, \hat{U}, \hat{\theta}, K\ (\hat{U}\hat{D}\hat{U}^T=\hat{P})$.

1) Calculează $f=U^T\Phi$; $v=Df$; $a^0=\lambda/a$.

2) Pentru $j=1, 2, \ldots, p$

   1) Pune $\hat{u}_{jj}=1$; $k_j=v_j$; $s^j=-f_j/a^{j-1}$.

   2) Actualizează $a^j=a^{j-1}+v_jf_j$;

   $\hat{d}_j=d_ja^{j-1}/(\lambda a^j)$.

3) Dacă $j > 1$, atunci pentru $i=1, \ldots, j-1$

$$\widehat{u}_{ij}=u_{ij}+s^j k_i ;$$

$$k_i : =k_i+v_j u_{ij}.$$

3) $\widehat{\theta}=\theta+K(y-\theta^T\Phi)^T/a^p$.

Observații

9) La sfîrșitul algoritmului 3 obținem $a^p=1/b=(\lambda+a\Phi^T P\Phi)/a$ și $K=UDU\Phi^T=P\Phi$ este vectorul de amplificare Kalman normalizat. În fond, a^0 a fost ales astfel încît $a^p=1/b$. Algoritmul poate fi ușor modificat pentru ca $\widehat{D}$, $\widehat{U}$, și $\widehat{\theta}$ să suprascrie D, U, și respectiv θ.

10) Algoritmul necesită numai $1{,}5p^2+(2m+4{,}5)\,p$ multiplicări.

11) Algoritmul 3 are bune proprietăți numerice; detalii despre etapa de actualizare a factorizării pot fi găsite în [23].

În Anexa 3 este listată subrutina UDCOV, pentru actualizarea factorizării U—D a matricei de covarianță. Rutina este programată pentru a necesita un minim de locații de memorie pentru date. UDCOV apelează subprogramul RFVPS pentru calculul produselor scalare; RFVPS a fost obținut prin adaptarea funcției SDOT din BLAS și utilizează cicluri desfășurate pentru mărirea eficienței calculelor.

Să considerăm acum relația (29) rescrisă astfel

$$\widehat{P}=UDU^T/\lambda-bKK^T. \tag{33}$$

Problema de actualizare se reduce la actualizarea factorizării U—D după o modificare de rang 1. Se poate aplica direct procedura A1 dată în Anexa 1, utilizînd subrutinele SROTMG și SROTM din pachetul BLAS. Totuși, această soluție nu este atrăgătoare din punct de vedere numeric, după cum se poate constata analizînd exemplul de mai jos, care constituie de fapt adaptarea exemplului 1 din secțiunea 3.

Exemplul 2: Fie $U=\mathrm{diag}\,(u_1, u_2)$, $D=\mathrm{diag}\,(d_1, d_2)$, $\Phi=[1\ \varepsilon]^T$, unde $d_i u_i^2=\sigma^2$, $i=1, 2$, $\sigma=1/\varepsilon$, și fl $(1+\varepsilon^2)=1$. Calcule elementare arată că, dacă se utilizează procedura A1, la a doua apelare a subrutinei SROTMG, datorită erorilor de rotunjire, matricea calculată H_1 are elementele $h_1'=-d_1u_1$, $h_2'=u_1\varepsilon^2$, astfel încît $\tau'=1+h_1'h_2'=1-d_1u_1^2\varepsilon^2=0$. Calculele nu mai pot fi deci continuate pentru nici o pereche de valori u_i, d_i satisfăcînd $d_iu_i^2=\sigma^2$ (vezi relația (A8) din Anexa 1). Deci, această tehnică nu poate fi în general recomandată. Toți ceilalți algoritmi discutați în articol au rezolvat cu succes această problemă-test.

Factorizarea tip rădăcină patrată

Punînd $S=UD^{1/2}$ în demonstrațiile de mai sus se poate obține ușor următorul algoritm de tip rădăcină pătrată:

Algoritmul 4; Actualizează θ utilizînd factorizarea SS^T a lui P.

Intrare: S, θ $(SS^T=P)$.

Ieșire: $\widehat{S}$, $\widehat{\theta}$, K $(\widehat{S}\widehat{S}^T=\widehat{P})$.

1) Calculează $f = S^T\Phi$; $a^0 = \lambda/a$.

2) Pentru $j = 1, 2, \ldots, p$

1) Pune $k_j = f_j s_{jj}$; $s^j = -f_j/a^{j-1}$.

2) Actualizează $a^j = a^{j-1} + f_j^2$;

$$d^j = (a^{j-1}/(\lambda a^j))^{1/2} ;$$

$$\widehat{s}_{jj} = d^j s_{jj}.$$

3) Dacă $j > 1$, atunci pentru $i = 1, \ldots, j-1$

$$\widehat{s}_{ij} = d^j (s_{ij} + s^j k_i) ;$$

$$k_i : = k_i + f_j s_{ij}.$$

3) $\widehat{\theta} = \theta + K(y - \theta^T\Phi)^T/a^p$.

Observații

12) Observația 9 de mai sus se poate transcrie aici.

13) Algoritmul necesită $2p^2 + (2m+5)p$ multiplicări și p rădăcini pătrate.

6. Rezultate experimentale

O parte din rezultatele teoretice prezentate în lucrare au fost aplicate în cadrul pachetului de programe ADCON [33, 34], implementat pe minicalculatoarele românești și destinat conducerii adaptive a proceselor tehnologice. ADCON include și alți algoritmi adaptivi [30] ; deocamdată este tratat numai cazul sistemelor monovariabile, dar numărul buclelor de reglare adaptivă este practic nelimitat.

Algoritmii de factorizare analizați în lucrare au fost programați și testați de autor, obținîndu-se rezultate bune în toate cazurile studiate. Nu s-a putut stabili experimental superioritatea din punct de vedere numeric a unui algoritm asupra altora. De remarcat, totuși, că algoritmul 2 este de preferat algoritmului 3, întrucît în implementare este evitată apariția depășirilor aritmetice.

Am studiat intensiv prin simulare pe un minicalculator CORAL 4011 comportarea algoritmilor prezentați în secțiunea 2. Pentru actualizarea factorizării a fost folosită subrutina UDCOV, listată în Anexa 3. Condițiile de simulare au fost alese astfel încît să fie apropiate celor întîlnite în mediu industrial. Zgomotul $\{w(t)\}$ a fost realizat ca semnal pseudoaleator cu distribuție normală, cu media $\bar{w}$ și abaterea standard σ. Pentru calculul mărimii de referință cu d pași înainte $(y^*(t+d))$ am folosit un model etalon. Introducerea acestuia a avut ca efect și atenuarea variațiilor mărimilor de comandă și de ieșire la modificarea bruscă a intrării de referință r (t). Semnalul $\{r(t)\}$ a fost ales de tip undă pătrată dreptunghiulară cu valoarea minimă 0 și valoarea maximă 1. În cazul algoritmului CMMPR1 eroarea de urmărire $y(t) - y^*(t)$ a fost

filtrată printr-un filtru specificat (vezi [4, 5]). Au fost efectuate numeroase teste pentru a studia influenţa condiţiilor iniţiale sau a unor parametri definitorii ai algoritmilor. În toate experimentele au fost calculate statistici elementare ale erorii de urmărire, atît în etapa tranzitorie de adaptare iniţială, cît şi în etapa finală. Analiza acestor statistici a confirmat proprietăţile bune de urmărire ale algoritmilor prezentaţi.

În continuare ilustrăm o parte din rezultatele obţinute pentru sistemul

$$(1-2{,}1q^{-1}+1{,}53q^{-2}-0{,}365q^{-3})\,y\,(t)=q^{-2}\,(1+0{,}4q^{-1})\,u\,(t)+$$

$$+(1-1{,}2q^{-1}+0{,}48q^{-2}-0{,}64q^{-3})\,w\,(t).$$

S-a utilizat modelul etalon definit de:

$$(1-1{,}9q^{-1}+1{,}23q^{-2}-0{,}265q^{-3})\,y^*\,(t)=q^{-2}\,(0{,}28+0{,}22\,q^{-1})\,r\,(t).$$

În fig. 1, 2 şi 3 sînt prezentate traiectoriile ieşirii sistemului şi modelului etalon rezultate folosind algoritmii CMMPR1, CMMPR2 şi respectiv, CMMPR3. În cazul algoritmului CMMPR1 s-a utilizat filtrul de eroare

$$1-1{,}2q^{-1}+0{,}48q^{-2}-0{,}64q^{-3}.$$

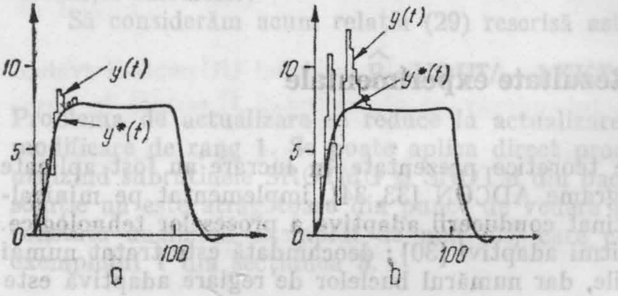


Fig. 1. Simulare utilizînd algoritmul CMMPR1.

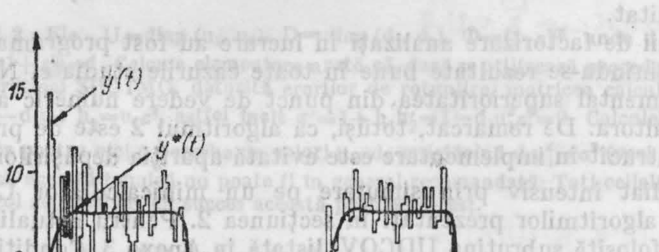


Fig. 2. Simulare utilizînd algoritmul CMMPR2.

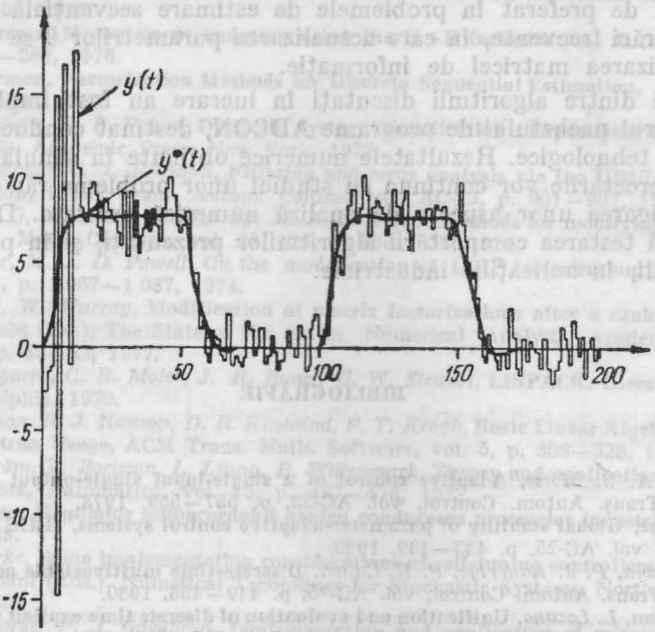


Fig. 3. Simulare utilizînd algoritmul CMMPR3.

În cazul algoritmilor CMMPR2 și CMMPR3 rezultatele au fost obținute în ipoteza $\overline{w}=0$, $\sigma=0,5$. În toate cazurile s-au adoptat valorile $\lambda\equiv1$ și $P(-1)=10I$. În fig. 1, a valorile inițiale ale parametrilor regulatorului adaptiv au fost alese apropiate de valorile lor optimale.

7. Concluzii

În lucrare sînt analizate tehnici de factorizare eficiente și atractive din punct de vedere numeric, aplicabile algoritmilor recursivi de tipul celor mai mici pătrate întîlniți în domeniul conducerii adaptive a sistemelor multivariabile deterministe sau stocastice. Sînt discutate într-o concepție unitară variante de tip rădăcină pătrată și de tip U—D pentru actualizarea factorizării matricei de informație sau a matricei de covarianță. Instrumentul matematic de bază utilizat îl constituie rotațiile plane, standard sau modificate. Sînt prezentate detalii de implementare și sînt date recomandări de utilizare a unor rutine matematice performante, recent elaborate, pentru rezolvarea problemelor de calcul asociate. Codurile de program listate în Anexe pot fi folosite și în probleme de estimare recursivă sau în unele probleme de minimizare. Din pricina limitării spațiului, nu s-au furnizat decît două subrutine de actualizare a factorizărilor.

Din punctul de vedere al efortului de calcul, algoritmii utilizînd matricea de informație sînt mai costisitori, în cazul multivariabil, decît ceilalți algoritmi.

Ei sînt însă de preferat în problemele de estimare secvențială cu estimații rare și măsurări frecvente, în care actualizarea parametrilor θ se face mai rar decît actualizarea matricei de informație.

O parte dintre algoritmii discutați în lucrare au fost implementați de autor în cadrul pachetului de programe ADCON, destinat conducerii adaptive a proceselor tehnologice. Rezultatele numerice obținute în simulare sînt încurajatoare. Cercetările vor continua cu studiul unor probleme rău condiționate și cu investigarea unor aspecte de analiză numerică asociate. De asemenea, este necesară testarea comportării algoritmilor prezentați, și în primul rînd a celor originali, în aplicațiile industriale.

BIBLIOGRAFIE

1. *A. Feuer, A. S. Morse*, **Adaptive control of a single-input single-output linear systems**, IEEE Trans. Autom. Control, vol. AC-23, p. 557—569, 1978.
2. *A. S. Morse*, **Global stability of parameter-adaptive control systems**, IEEE Trans. Autom. Control, vol. AC-25, p. 433—439, 1980.
3. *G. C. Goodwin, P. J. Ramadge, P. E. Caines*, **Discrete-time multivariable adaptive control**, IEEE Trans. Autom. Control, vol. AC-25, p. 449—456, 1980.
4. *I. D. Landau, L. Lozano*, **Unification and evaluation of discrete time explicit model reference adaptive control designs**, Automatica, vol. 17, p. 593—611, 1981.
5. *L. Lozano, I. D. Landau*, **Redesign of explicit and implicit discrete time model reference adaptive control systems**, Int. J. Control, vol. 33, p. 247—268, 1981.
6. *L. Lozano*, **Adaptive control with forgetting factor**, In Prepr. of the 8-th IFAC World Congress, Kyoto, Japan, vol. VII, p. 83—88, 1981.
7. *G. C. Goodwin, K. S. Sin, K. K. Saluja*, **Stochastic adaptive control and prediction — The general delay-colored noise case**, IEEE Trans. Autom. Control, vol. AC-25, p. 946—950, 1980.
8. *G. C. Goodwin, P. J. Ramadge, P. E. Caines*, **A globally convergent adaptive predictor**, Automatica, vol. 17, p. 135—140, 1981.
9. *G. C. Goodwin, P. J. Ramadge, P. E. Caines*, **Discrete time stochastic adaptive control**, SIAM J. Control Optimiz., vol. 19, p. 829—853, 1981.
10. *K. S. Sin, G. C. Goodwin. R. R. Bitmead*, **An adaptive d-step ahead predictor based on least squares**, IEEE Trans. Autom. Control, Vol. AC-25, p. 1161—1165, 1980.
11. *K. S. Sin, G. C. Goodwin*, **Stochastic adaptive control using a modified least squares algorithm**, Techn. Rep. EE7909, Sec. Rev., Dep. Electr. Engng., Univ. of Newcastle, Australia, 1981.
12. *J. B. Moore, G. Ledwich*, **Multivariable adaptive parameter and state estimators with convergence analysis**, J. Austr. Math. Soc., vol. 21, Ser. B, p. 176—197, 1979.
13. *J. B. Moore, R. Kumar*, **Convergence of weighted minimum variance N-step ahead prediction/control schemes**, Proc. of the 19-th IEEE Conf. Decision and Control, Albuquerque, NM, p. 968—973, 1980.
14. *R. Kumar, J. B. More*, **Convergence of adaptive minimum variance algorithms via weighting coefficient selection**, IEEE Trans. Autom. Control, Vol. AC-27, p. 146—153, 1982.
15. *V. Sima*, **Contribuții la realizarea sistemelor adaptive cu model etalon**, Teză de doctorat, IPB, 1982.
16. *V. Sima*, **A sufficient condition for convergence of recursive least squares adaptive control algorithms**, Prepr. 27 Intern. Wiss. Kolloquium, Techn. Hoch. Ilmenau, GDR, oct., 1982.
17. *V. Sima*, **On recursive least squares adaptive control algorithms**, Proc. of the 1-st IASTED Int. Symp. and Course „Measurement and Control", Tunis, Tunisia, sept. 1—3, 1982.
18. *V. Sima*, **Rezultate noi în conducerea adaptivă**, Prepr. A III-a Sesiune „Modelarea, simularea, identificarea și optimizarea proceselor tehnologice". Univ. Galați, 29—30 oct., 1982.

19. *V. Peterka*, **A square-root filter for a real time multivariable regression**, Kybernetika, vol. 11, p. 53, 1975.
20. *G. J. Bierman*, **Measurement updating using the U — D factorization**, Automatica, vol. 12, p. 375—382, 1976.
21. *G. J. Bierman*, **Factorization Methods for Discrete Sequential Estimation**, Academic Press, New York, 1977.
22. *G. C. Goodwin, R. L. Payne*, **Dynamic System Identification : Experiment Design and Data Analysis**, Academic Press, New York, 1977.
23. *C. L. Thornton, G. J. Bierman*, **Filtering and error analysis via the UDU^T covariance factorization**, IEEE Trans. Autom. Control, vol. AC-23, p. 901—907, 1978.
24. *P. E. Gill, G. H. Golub, W. Murray, M. A. Saunders*, **Methods for modifying matrix factorizations**, Math. Comput., vol. 28, 126, p. 505—535, 1974.
25. *R. Fletcher, M. J. D. Powell*, **On the modification of LDL^T factorization**, Math. Comput., vol. 28, p. 1 067—1 087, 1974.
26. *P. E. Gill, W. Murray*, **Modification of matrix factorizations after a rank-one change**, În D. Jacobs (Ed.), The State of the Art in Numerical Analysis, Academic Press, New York, p. 55—83, 1977.
27. *J. J. Dongarra, C. B. Moler, J. R. Bunch G. W. Stewart*, **LINPACK. Users' Guide**, SIAM, Philadelphia, 1979.
28. *C. L. Lawson, R. J. Hanson, D. R. Kincaind, F. T. Krogh*, **Basic Linear Algebra Subprograms for Fortran Usage**, ACM Trans. Math. Software, vol. 5, p. 308—323, 1979.
29. *K. J. Aström, V. Borisson, L. Ljung, B. Wittenmark*, **Theory and applications of self-tuning regulators**, Automatica, vol. 13, p. 457—476, 1977.
30. *Th. Popescu*, **Regulator autoacordabil pentru conducerea proceselor industriale**, AMC, vol. 35, 1983.
31. *D. V. Clarke*, **Some implementation considerations of self-tuning controllers**, În F. Archetti, M. Cugiani (Eds.), Numerical Techniques for Stochastic Systems, North Holland Publ. Co., 1980.
32. *D. W. Clarke, P. J. Gawthrop*, **Implementation and application of microprocessors-based self-tuners**, Automatica, vol. 17, p. 233—244, 1981.
33. *V. Sima, Th. D. Popescu*, ADCON — **Adaptive control package**, În G. Ferrate, E. A. Puente (Eds.), Software for Computer Control 1982, Proc. of 3-rd IFAC/IFIP Symp., Madrid, 5—8 oct., 1982.
34. *Th. D. Popescu, V. Sima*, **Experiments with an adaptive control package**, Prepr. Applied Control and Identification, Copenhaga, 28 iunie, 1 iulie, 1983.

ANEXA 1

Actualizarea factorizării U—D după o modificare de rang 1

Să considerăm factorizarea U—D a unei matrice pozitiv definite $P \in R^{p \times p}$, adică $P = UDU^T$, unde $U = (u_{ij})$ este o matrice superior triunghiulară, iar $D = \mathrm{diag}(d_1, \ldots, d_p)$ este o matrice diagonală pozitiv definită. Dorim să calculăm, stabil şi eficient, factorizarea U—D a matricei

$$\hat{P} = P + axx^T \tag{A1}$$

unde $x \in R^p$ este un vector dat, iar $a \in R$ este un scalar dat, presupus iniţial a fi pozitiv. Scriind (A1) astfel

$$\hat{P} = UDU^T + axx^T = [UD^{1/2} \quad a^{1/2}x]\, G^T G \begin{bmatrix} D^{1/2}U^T \\ a^{1/2}x^T \end{bmatrix} \tag{A2}$$

şi alegînd matricea ortogonală G, aşa încît

$$G \begin{bmatrix} D^{1/2}U^T \\ a^{1/2}x^T \end{bmatrix} = \begin{bmatrix} \hat{D}^{1/2}\hat{U}^T \\ 0 \end{bmatrix} \tag{A3}$$

unde $\hat{U}$ este matrice superior triunghiulară, iar $\hat{D}$ este matrice diagonală pozitiv definită, obţinem $\hat{P} = \hat{U}\hat{D}\hat{U}^T$. Matricea G poate fi determinată ca produs a p rotaţii plane, $G = G_1 G_2 \ldots G_p$, rotaţia G_j, $j = p, p-1, \ldots, 1$, fiind aplicată în planul $(j, p+1)$. G_j este definită de cosinusul $c_j = d_j^{1/2} u_{jj}/r$ şi sinusul $s_j = (a^j)^{1/2} x_j^j / r$, unde $r = \pm(u_{jj}^2 + (x_j^j)^2)^{1/2}$, iar indicele superior j denotă valoarea variabilelor la a j-etapă de calcul ($a^p = a$, $x^p = x$). Pentru a evita operaţiile rădăcină pătrată, se pot utiliza rotaţii plane modificate, ca în [28]. Ideea de bază a metodei este de a refactoriza produsul $g_j \,\mathrm{diag}\,[d_j^{1/2} (a^j)^{1/2}]$ astfel

$$\begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix} \begin{bmatrix} d_j^{1/2} & 0 \\ 0 & (a^j)^{1/2} \end{bmatrix} = \begin{bmatrix} \hat{d}_j^{1/2} & 0 \\ 0 & (a^{j-1})^{1/2} \end{bmatrix} H_j \tag{A4}$$

unde matricea H_j de tip 2×2 are în general două elemente egale cu 1. Atunci,

$$\begin{bmatrix} c_j & s_j \\ -s_j & c_j \end{bmatrix} \begin{bmatrix} d_j^{1/2} & 0 \\ 0 & (a^j)^{1/2} \end{bmatrix} \begin{bmatrix} u_{1j} \ldots u_{jj} \\ x_1^j \ldots x_j^j \end{bmatrix} = \begin{bmatrix} \hat{d}_j^{1/2} & 0 \\ 0 & (a^{j-1})^{1/2} \end{bmatrix} H_j \begin{bmatrix} u_{1j} \ldots u_{jj} \\ x_1^j \ldots x_j^j \end{bmatrix} =$$

$$= \begin{bmatrix} \hat{d}_j^{1/2} & 0 \\ 0 & (a^{j-1})^{1/2} \end{bmatrix} \begin{bmatrix} \hat{u}_{1j} \ldots \hat{u}_{j-1,j} & \hat{u}_{jj} \\ x_1^{j-1} \ldots x_{j-1}^{j-1} & 0 \end{bmatrix} \tag{A5}$$

Vom descrie pe scurt calculul lui H_j. Pentru simplitate, indicii pentru c_j şi s_j vor fi omişi. Presupunînd că $|s| < |c|$, putem scrie

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} d_j^{1/2} & 0 \\ 0 & (a^j)^{1/2} \end{bmatrix} = \begin{bmatrix} d_j^{1/2}c & 0 \\ 0 & (a^j)^{1/2}c \end{bmatrix} \begin{bmatrix} 1 & t\,(a^j/d_j)^{1/2} \\ -t\,(d_j/a^j)^{1/2} & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \hat{d}_j^{1/2} & 0 \\ 0 & (a^{j-1})^{1/2} \end{bmatrix} \begin{bmatrix} 1 & a^j x_j^j/(d_j u_{jj}) \\ -x_j^j/u_{jj} & 1 \end{bmatrix} \tag{A6}$$

unde $\hat{d}_j = d_j c^2$, $a^{j-1} = a^j c^2$, iar $t = s/c = (a^j/d_j)^{1/2} x^j_j / u_{jj}$. Notînd $h_1 = a^j x^j_j/(d_j u_{jj})$ și $h_2 = x^j_j/u_{jj}$, obținem $1/c^2 = 1 + t^2 = 1 + h_1 h_2$. Deci,

$$\tau \stackrel{\Delta}{=} 1 + h_1 h_2,\ \hat{d}_j = d_j/\tau,\ a^{j-1} = a^j/\tau,\ H_j = \begin{bmatrix} 1 & h_1 \\ -h_2 & 1 \end{bmatrix} \tag{A7}$$

Din motive de stabilitate numerică, pentru $|s| \geqslant |c|$, se utilizează alte relații care se deduc similar. Notînd $h_1' = d_j u_{jj}/(a^j x^j_j)$, $h_2' = u_{jj}/x^j_j$, avem

$$\tau' \stackrel{\Delta}{=} 1 + h_1' h_2',\ \hat{d}_j = a^j/\tau',\ a^{j-1} = d_j/\tau',\ H_j = \begin{bmatrix} h_1' & 1 \\ -1 & h_2' \end{bmatrix} \tag{A8}$$

Pentru a evita depășirile aritmetice superioare și inferioare, $\hat{d}_j$ și a^{j-1} sînt menținute între anumite limite prestabilite, prin scalarea problemei, dacă este necesar. În aceste rare cazuri, în care se face scalare, matricea H_j poate avea mai mult de două elemente diferite de 1.

Întrucît în calculele efective nu apar rădăcini pătrate, tehnica se poate aplica și în cazul $a \lhd 0$.

Pentru determinarea rotațiilor plane modificate descrise mai sus se poate utiliza subrutina SROTMG din pachetul BLAS [28]. Actualizarea factorizării U—D este realizată așadar de următoarea procedură.

Procedura A1: Actualizează factorizarea U—D după adăugarea unei diade.

1) Pentru $j = p, p-1, \ldots, 1$

   1) Utilizează SROTMG cu datele d_j, a, u_{jj} și x_j pentru a determina H_j și pentru a actualiza d_j, a și u_{jj}.

   2) Pentru $j > 1$

$$\begin{bmatrix} u_{1j} \ldots u_{j-1,j} \\ x_1 \ \ldots \ x_{j-1} \end{bmatrix} := H_j \begin{bmatrix} u_{1j} \ldots u_{j-1,j} \\ x_1 \ \ldots \ x_{j-1} \end{bmatrix}$$

Observații

A1) Unele variabile sînt suprascrise în procedură. În final, U și D conțin cantitățile actualizate $\hat{U}$ și, respectiv, $\hat{D}$.

A2) Procedura necesită $p^2 + 10p$ multiplicări, dacă nu este necesară nici o scalare.

A3) Tehnica poate fi ușor adaptată factorizării LDL^T.

Încheiem această secțiune prin deducerea pe o cale nouă a algoritmului din Anexa II din [23]. În acest caz impunem ca U și $\hat{U}$ să fie superior triunghiulare unitate. Să considerăm cazul $|s| \lhd |c|$, deci H_j este definit ca în (A7). Atunci avem $\hat{u}_{jj} = u_{jj} + a^j (x^j_j)^2/(d_j u_{jj}) = 1 + t^2$. Pentru a obține $u_{jj} = 1$, coloana a j-a din matricea $\hat{U}$, rezultată din (A5), trebuie împărțită prin $1 + t^2 (= d_j/\hat{d}_j)$, deci $\hat{d}_j$ trebuie multiplicat cu $(1 + t^2)^2$, pentru a conserva factorizarea. După calcule elementare, se obține :

$$\hat{d}_j = d_j + a^j (x^j_j)^2 \tag{A9.a}$$

$$a^{j-1} = a^j d_j / \hat{d}_j \tag{A9.b}$$

$$\hat{u}_{ij} = (d_j u_{ij} + \cdot x^j_j x^j_i)/\hat{d}_j = u_{ij} + a^j x^j_j x^{j-1}_i/\hat{d}_j \tag{A9.c}$$

$$x^{j-1}_i = x^j_i = u_{ij} x^j_j. \tag{A9.d}$$

Acestea sînt tocmai relațiile utilizate în algoritmul din [23]. Pentru actualizarea lui u_{ij} se folosește una sau alta dintre cele două variante de mai sus în funcție de valoarea raportului $d_j/\hat{d}_j$. Relațiile prezentate se obțin și în cazul $|s| \geqslant |c|$.

ANEXA 2

Actualizarea factorizării U*U' a matricei de informaţie

```
C       SUBRUTINA SQRINF
C
C SQRINF  ACTUALIZEAZA FACTORIZAREA  U*U'  A MATRICEI DE INFORMATIE,
C UNDE  U ESTE SUPERIOR TRIUNGHIULARA, DUPA ADAUGAREA UNEI DIADE:
C       (ALAMBD)*U*U' + A*(PHI)*(PHI)'.
C SIMULTAN, SQRINF ACTUALIZEAZA CORESPUNZATOR PERECHEA (Z,Y),
C UNDE  Z  ESTE UN VECTOR, IAR  Y  ESTE UN SCALAR.
C
C MOD DE UTILIZARE
C CALL SQRINF (IP,U,PHI,Z,Y,A,ALAMBD)
C
C DESCRIEREA PARAMETRILOR
C IP     - ORDINUL MATRICEI  U  SI DIMENSIUNEA VECTORILOR  PHI, Z.
C U      - VECTOR DAT CU  IP*(IP+1)/2  ELEMENTE CONTININD PARTEA
C          TRIUNGHIULARA SUPERIOARA A MATRICEI U, PE COLOANE.
C          ESTE ACTUALIZAT IN RUTINA.
C PHI    - VECTOR DAT CU  IP  ELEMENTE. ESTE ALTERAT IN RUTINA.
C Z      - VECTOR DAT CU  IP  ELEMENTE. ESTE ACTUALIZAT IN RUTINA.
C Y      - SCALAR DAT. ESTE ACTUALIZAT IN RUTINA.
C A      - SCALAR DAT.
C ALAMBD - SCALAR DAT.
C
C SUBPROGRAME UTILIZATE
C SROTG
C
C METODA
C SE DETERMINA ROTATII GIVENS PENTRU A ANULA ELEMENTELE ULTIMEI
C LINII DIN MATRICEA:
C
C      [ SQRT(ALAMBD) * U' ]
C      [   SQRT(A) * (PHI)' ]
C
C IN ORDINEA  J = IP,IP-1,...,1. ROTATIILE SE APLICA ACESTEI MATRICI
C SI VECTORULUI
C
C      [ SQRT(ALAMBD) * Z ]
C      [     SQRT(A) * Y   ].
C
        SUBROUTINE SQRINF (IP,U,PHI,Z,Y,A,ALAMBD)
C
        DIMENSION U(1),PHI(1),Z(1)
C
        REAL C,C1,S,S1,SQRTA,SQRTLA,TEMP
        INTEGER I,J,JI,JJ,JM1,K,L
        LOGICAL FORGET
C
        SQRTLA = SQRT (ALAMBD)
        SQRTA = SQRT (A)
        FORGET = .TRUE.
        IF (ALAMBD .EQ. 1.0) FORGET = .FALSE.
C
        IF (A .EQ. 1.0) GO TO 10
           Y = Y * SQRTA
           DO 5  J = 1,IP
              PHI(J) = PHI(J) * SQRTA
    5      CONTINUE
C
   10   CONTINUE
        JJ = IP * (IP + 1) / 2
C
        DO 40 L = 1,IP
           J = IP - L + 1
           JI = JJ - J
           IF (PHI(J) .EQ. 0.0) GO TO 30
```

```
            IF (FORGET) U(JJ) = U(JJ) * SQRTLA
            CALL SROTG (U(JJ),PHI(J),C,S)
C
            C1 = C
            S1 = S
            IF (.NOT. FORGET) GO TO 15
              C1 = C1 * SQRTLA
              S1 = S1 * SQRTLA
   15       CONTINUE
            JM1 = J - 1
            IF (JM1 .EQ. 0) GO TO 25
C
              K = JI
              DO 20 I = 1,JM1
                K = K + 1
                TEMP = U(K)
                U(K) = C1 * TEMP + S * PHI(I)
                PHI(I) = C * PHI(I) - S1 * TEMP
   20         CONTINUE
C
   25       CONTINUE
            TEMP = Z(J)
            Z(J) = C1 * TEMP + S * Y
            Y = C * Y - S1 * TEMP
C
   30     CONTINUE
          JJ = JI
   40 CONTINUE
C
      RETURN
      END
```

ANEXA 3

Actualizarea factorizării U—D a matricei de covarianță

```
C        SUBRUTINA UDCOV
C
C UDCOV  ACTUALIZEAZA FACTORIZAREA  U*D*U'  A MATRICEI DE COVARIANTA,
C UNDE  U  ESTE SUPERIOR TRIUNGHIULARA UNITATE, IAR  D  ESTE DIAGONALA,
C SI DETERMINA VECTORUL DE AMPLIFICARE KALMAN NORMALIZAT.
C
C MOD DE UTILIZARE
C CALL UDCOV (IP,U,PHI,A,ALAMBD,AK,AP)
C
C DESCRIEREA PARAMETRILOR
C IP     - ORDINUL MATRICILOR  U, D  SI DIMENSIUNEA VECTORILOR  PHI, AK.
C U      - VECTOR DAT CU  IP*(IP+1)/2  ELEMENTE CONTININD PARTEA
C          TRIUNGHIULARA SUPERIOARA STRICTA A MATRICEI  U, PE COLOANE.
C          IN LOCATIILE CORESPUNZATOARE ELEMENTELOR DIAGONALE UNITATE
C          NEMEMORATE ALE MATRICEI  U,  SE GASESC ELEMENTELE DIAGONALE
C          ALE MATRICEI D. ESTE ACTUALIZAT IN RUTINA.
C PHI    - VECTOR DAT CU  IP  ELEMENTE.
C A      - SCALAR DAT.
C ALAMBD - SCALAR DAT.
C AK     - VECTOR REZULTAT CU  IP  ELEMENTE CONTININD AMPLIFICAREA
C          KALMAN NORMALIZATA.
C AP     - SCALAR REZULTAT CONTININD INVERSUL FACTORULUI DE NORMALIZARE.
C          (AMPLIFICAREA KALMAN ESTE  AK / AP).
C
C SUBPROGRAME UTILIZATE
C RFVPS
C
C METODA
C SE UTILIZEAZA ALGORITMUL DE ACTUALIZARE A FACTORIZARII U-D A MATRICEI
C DE COVARIANTA DESCRIS IN [1].
C
C REFERINTE BIBLIOGRAFICE
C [1] THORNTON, C.L., BIERMAN, G.J., FILTERING AND ERROR ANALYSIS
C                             T
C     VIA THE  UDU   COVARIANCE FACTORIZATION, IEEE TRANS. AUT.
C     CONTR., AC-23, 901-907, 1978.
C
      SUBROUTINE UDCOV (IP,U,PHI,A,ALAMBD,AK,AP)
C
      DIMENSION U(1),PHI(1),AK(1)
C
      REAL S,F,AJ,AJM1,AKJ,TEMP
      INTEGER I,J,K,KD,JM1
      LOGICAL FORGET
C
      FORGET = .TRUE.
      IF (ALAMBD .EQ. 1.0) FORGET = .FALSE.
C
      K = 0
      KD = 0
      AJ = ALAMBD/A
C
      DO 20 J = 1,IP
         K = K + 1
         KD = KD + J
         JM1 = J - 1
         F =  PHI(J)
         IF (JM1 .GT. 0) F = F + RFVPS (JM1,U(K),PHI)
         AKJ = U(KD) * F
         AK(J) = AKJ
         AJM1 = AJ
         AJ = AJ + AKJ * F
         S = AJ
         IF (FORGET) S = S * ALAMBD
```

```
      U(KD) = U(KD) * AJM1 / S
      IF (JM1 .EQ. 0) GO TO 20
C
      S = - F / AJM1
      DO 10 I = 1,JM1
         TEMP = U(K)
         U(K) = TEMP + S * AK(I)
         AK(I) = AK(I) + AKJ * TEMP
         K = K + 1
   10 CONTINUE
C
   20 CONTINUE
C
      AP = AJ
C
      RETURN
      END
C
C
C
C     FUNCTIA RFVPS
C
C  RFVPS CALCULEAZA PRODUSUL SCALAR A DOI VECTORI.
C
C  MOD DE UTILIZARE
C  X = RFVPS (N,SX,SY)
C
C  DESCRIEREA PARAMETRILOR
C  N      - DIMENSIUNEA VECTORILOR  SX SI SY.
C  SX     - VECTOR DAT CU  N  ELEMENTE.
C  SY     - VECTOR DAT CU  N  ELEMENTE.
C  RFVPS  - SUMA REZULTATA A PRODUSELOR  SX(I)*SY(I) ,  I = 1,...,N.
C
C  METODA
C  SE UTILIZEAZA CICLURI DESFASURATE, PENTRU MARIREA EFICIENTEI.
C  ESTE ADAPTAT SUBPROGRAMUL  SDOT  DIN PACHETUL  BLAS [1].
C
C  REFERINTE BIBLIOGRAFICE
C  [1] LAWSON, C.L., R.J. HANSON,D.R.KINCAID, F.T. KROGH. BASIC
C      LINEAR ALGEBRA SUBPROGRAMS FOR FORTRAN USAGE. ACM TRANS.
C      MATH. SOFTWARE, 5, 308-323, 1979.
C
      REAL FUNCTION RFVPS(N,SX,SY)
C
      INTEGER                N
      REAL                   SX(1),SY(1)
C
      INTEGER                I,M,MP1
C
      RFVPS= 0.0E0
      IF (N.LE.0) RETURN
C
   15 M = N-(N/5)*5
      IF (M.EQ.0) GO TO 25
      DO 20 I=1,M
         RFVPS= RFVPS+SX(I)*SY(I)
   20 CONTINUE
      IF (N.LT.5) RETURN
   25 MP1 = M+1
      DO 30 I=MP1,N,5
         RFVPS= RFVPS+SX(I)*SY(I)+SX(I+1)*SY(I+1)+SX(I+2)*SY(I+2)+SX(I
     1   +3)*SY(I+3)+SX(I+4)*SY(I+4)
   30 CONTINUE
      RETURN
      END
```

METODE DE ESTIMARE RECURSIVĂ A PARAMETRILOR SISTEMELOR

Dr. ing. *Th. D. Popescu*
I.T.C.I.

1. Introducere

În multe aplicaţii se urmăreşte estimarea parametrilor sistemelor în mod *recursiv*, pe măsura evoluţiei acestora. Acest mod de abordare prezintă interes, în special pentru actualizarea continuă a modelului sistemului, cînd caracteristicile acestuia şi/sau ale mediului se modifică în timp. Tehnicile de estimare recursivă au largi aplicaţii în modelarea şi predicţia seriilor de timp, estimarea stării, conducerea adaptivă a proceselor, recunoaşterea formelor. Acestea utilizează datele pe măsura culegerii lor, fără a necesita întreg setul de date intrare-ieşire din proces şi pot fi uşor adaptate pentru estimarea în timp real a parametrilor sistemelor.

Metodele de estimare recursivă permit o formulare secvenţială a problemei de estimare şi necesită un volum de memorie şi de calcul redus faţă de metodele de estimare nerecursivă, totuşi din punct de vedere al preciziei sînt inferioare acestora din urmă. Ele asigură o viteză de convergenţă convenabilă, în măsura în care se cunoaşte a priori domeniul de variaţie al parametrilor procesului, se iniţializează corect anumite matrici de ponderare şi se aleg corespunzător valorile iniţiale ale parametrilor modelului.

În cadrul estimării recursive a parametrilor sistemelor se pot utiliza următoarele clase de metode :

— metode utilizînd un model de referinţă ;

— metode de aproximare stochastică ;

— metode recursive.

Metodele utilizînd un model de referinţă (Landau, 1974) se folosesc, aproape exclusiv, în cazul sistemelor deterministe, iar în analiza acestora se foloseşte în principal metoda Liapunov (Parks, 1966 ; Narendra şi Kudva, 1974) şi teoria hiperstabilităţii (Landau, 1974). Multe din metodele de estimare, utilizînd un model de referinţă, au bune proprietăţi în cazul determinist, dar nu se comportă întotdeauna satisfăcător în prezenţa zgomotului ; sînt totuşi adecvate estimării parametrilor variabili în timp.

Metodele de aproximare stochastică (Saridis-Stein, 1968 ; Saridis, 1974 ; Sakrison, 1967), spre deosebire de schemele de estimare deterministe utilizînd un model de referinţă, au fost dezvoltate în special pentru estimarea parametrilor în prezenţa zgomotului.

Viteza de convergență a algoritmilor de aproximare stochastică poate fi, totuși, destul de redusă în cazul unei alegeri neinspirate a secvenței factorilor de amplificare.

Metodele recursive reprezintă acele metode de estimare care utilizează construcții de tipul celor mai mici pătrate recursive (Aström-Eykhoff, 1971 ; Eykhoff, 1974 ; Furth, 1973 ; Gertler-Banyasz, 1974 ; Hastings James-Sage 1974 ; Söderström și colab., 1974, 1978 ; Young, 1968, 1970, 1974, 1976 etc.). Analiza acestor metode se poate face utilizînd tehnica propusă de Ljung (1977a, 1977b), care face apel la o ecuație diferențială ordinară asociată algoritmilor de estimare recursivă (se utilizează apoi metodele Liapunov), sau tehnica propusă de Hannan (1976), care reprezintă o aplicare sofisticată a teoriei probabilităților, analiza reducîndu-se la studiul unei funcții deterministe de tip Liapunov. Metodele recursive, în general, au bune proprietăți de convergență și se comportă bine în prezența zgomotului.

În cadrul acestei lucrări sînt analizate și comparate principalele metode de estimare recursivă tratate în literatură, accentul fiind pus în special pe metodele din ultima categorie. Aceste metode sînt prezentate în secțiunea 3. Metodele prezintă elemente comune și sînt tratate într-un cadru unitar ; sînt prezentate de asemenea și unele modificări ce pot fi aduse algoritmilor recursivi avînd drept scop : obținerea versiunilor în timp real, îmbunătățirea vitezei de convergență. În secțiunea 4 sînt prezentate instrumentele de analiză a algoritmilor recursivi și se face o sinteză a rezultatelor acestei analize. Aspectele legate de implementarea acestor algoritmi constituie obiectul secțiunii 5 a acestei lucrări.

Metodele de estimare recursivă ce se prezintă au constituit obiectul unui număr mare de simulări. Rezultatele numerice obținute în estimarea parametrilor unor sisteme funcționînd în buclă deschisă și în estimarea parametrilor unor regulatoare autoacordabile, în cadrul unor sisteme funcționînd în buclă închisă, sînt prezentate în cadrul altor lucrări (Popescu, 1980, 1981).

2. Preliminarii

În cadrul acestei secțiuni sînt prezentate metode de estimare recursivă frecvent utilizate în practică și care au elemente comune :

— metoda celor mai mici pătrate recursivă — RCMMP ;

— metoda celor mai mici pătrate generalizată recursivă — RCMMPG ;

— metoda variabilelor instrumentale recursivă — RVI ;

— metoda celor mai mici pătrate extinsă recursivă (metoda matricei extinse, metoda verosimilității maxime aproximative, metoda Panuska) — RCMMPE ;

— metoda verosimilității maxime recursivă — RVM.

Multe din aceste metode se regăsesc în literatură combinate (ex. RVI și RCMMPE — Young 1974), sau sub alte denumiri. Metoda RCMMP este bine cunoscută, fiind tratată de mulți autori (ex. Aström și Eykhoff, 1971). Primele lucrări asupra metodei RVI au fost elaborate de Mayne (1967), Wong

şi Polak (1967) şi Young (1968). Metoda RCMMPG a fost sugerată de Hastings James şi Sage (1969) şi se bazează pe versiunea off-line a metodei (Clarke, 1967). Metoda RCMMPE a fost propusă de Panuska (1968, 1969) şi Young (1968), iar prezentarea metodei RVM este făcută de Furth (1973) şi Söderström (1973). Alte referinţe asupra acestor metode sînt date de Söderström, Ljung şi Gustavsson (1974), Tertişco şi Stoica (1980).

Aşa cum se va arăta în continuare, toţi aceşti algoritmi au aceeaşi structură. Pentru tratarea lor într-un cadru general vor fi făcute cîteva ipoteze şi convenţii generale.

Fie u (1), ..., u (N) semnalul de intrare şi y (1), ..., y (N) semnalul de ieşire pentru care se determină, cu metodele amintite, un model avînd următoarea structură generală:

$$\hat{A}(q^{-1})\, y(t) = \hat{B}(q^{-1})\, z(t) + \hat{H}(q^{-1})\, \varepsilon(t) \tag{2.1}$$

unde $\varepsilon(t)$ reprezintă reziduul, iar $\hat{A}(q^{-1})$ şi $\hat{B}(q^{-1})$ sînt polinoame în operatorul de întîrziere q^{-1} $(q^{-1}x(t) = x(t-1))$:

$$\hat{A}(q^{-1}) = 1 + \hat{a}_1 q^{-1} + \ldots + \hat{a}_{n_a} q^{-n_a}$$

$$\hat{B}(q^{-1}) = b_1 + \hat{b}_2 q^{-1} + \ldots + \hat{b}_{n_b} q^{-n_b}$$

Filtrul $\hat{H}(q^{-1})$ are forme diferite, funcţie de metoda utilizată, astfel:

Pentru RCMMP: $$\hat{H}(q^{-1}) = 1 \tag{2.2}$$

Pentru RIV, $\hat{H}(q^{-1})$ nu este specificat; din motive formale se va utiliza tot modelul (2.2), reziduul $\varepsilon(t)$ fiind arbitrar.

Pentru RCMMPG: $$\hat{H}(q^{-1}) = 1/\hat{C}(q^{-1}) = 1/(1 + \hat{c}_1 q^{-1} + \ldots + \hat{c}_{n_c} q^{-n_c}) \tag{2.3}$$

iar pentru RCMMPE şi RVM: $$\hat{H}(q^{-1}) = \hat{C}(q^{-1}) = 1 + \hat{c}_1 q^{-1} + \ldots + \hat{c}_{n_c} q^{-n_c} \tag{2.4}$$

Unele dintre metode ca, de exemplu, RCMMPE (Talmon şi van den Boom, 1973) şi RVM pot fi utilizate, de asemenea, şi pentru alte structuri ale modelului.

Includerea în structura modelului (2.1) a mai multor semnale de intrare şi a timpului mort nu va ridica, în general, probleme deosebite. Din motive de simplificare a notaţiilor va fi tratat numai cazul sistemelor cu o singură intrare, fără timp mort.

Deoarece sîntem interesaţi în special în analiza proprietăţilor de consistenţă ale algoritmilor se va presupune că, pentru fiecare din metode, sistemul are o formă corespunzătoare structurii modelului utilizat; prin urmare se va presupune că sistemul este de forma:

$$A(q^{-1})\, y(t) = B(q^{-1})\, u(t) + H(q^{-1})\, e(t) \tag{2.5}$$

unde e (t) este zgomot alb, iar polinoamele $A(q^{-1})$ şi $B(q^{-1})$ sînt date de relaţiile:

$$A(q^{-1}) = 1 + a_1 q^{-1} + \ldots + a_{n_a} q^{-n_a}$$

$$B(q^{-1}) = b_1 + b_2 q^{-1} + \ldots + b_{n_b} q^{-n_b}$$

Pentru anumite interpretări, caracteristice metodei verosimilității maxime, se presupune de asemenea că e (t) este gaussian, dar această ipoteză nu este esențială pentru analiza de convergență.

Filtrul $H(q^{-1})$ se presupune a avea următoarele structuri:

$$\text{Pentru RCMMP}: H(q^{-1})=1 \tag{2.6}$$

$$\text{Pentru RCMMPG}: H(q^{-1})=1/C(q^{-1})=1/(1+c_1q^{-1}+\ldots+c_{n_c}q^{-n_c}) \tag{2.7}$$

$$\text{iar pentru RCMMPE și RVM}: H(q^{-1})=C(q^{-1})=1+c_1q^{-1}+\ldots+c_{n_c}q^{-n_c} \tag{2.8}$$

Pentru cele mai multe din variantele metodei RVI nu este necesar a se specifica structura filtrului $H(q^{-1})$. Se presupune că ordinul sistemului este cunoscut a priori.

În general se presupune că descrierea sistemului (2.5) este astfel încît perechea de polinoame A (z), B (z) (z fiind o variabilă complexă arbitrară, înlocuind q^{-1}) nu are nici un factor comun ; aceeași ipoteză se face și pentru polinoamele A (z) și C (z) în cazul metodelor RCMMPE și RVM. Această ipoteză implică în special controlabilitatea procesului. De asemenea, se presupune că sistemul este asimptotic stabil. Pentru sistemul operînd în buclă deschisă, aceasta înseamnă că polinomul A (z) are toate zerourile în afara cercului unitate ; aceeași ipoteză se face și pentru polinomul C (z), ceea ce reprezintă o restricție minoră (teorema de factorizare spectrală, Aström 1970) și este necesară pentru a se asigura inversarea modelului zgomotului (RCMMPE și RVM), deci dispersia finită a reziduurilor.

Semnalul de intrare poate fi determinat în mai multe moduri, rezultatele ce se prezintă fiind valabile pentru o intrare de tip excitație persistentă (Aström și Bohlin, 1965), de exemplu zgomot alb filtrat, situație ce apare destul de frecvent. În special, analiza acoperă cazul funcționării în buclă închisă a sistemului, în care regulatorul este variabil în timp și se determină utilizînd modelul curent al procesului (cazul sistemelor adaptive).

3. Metode de estimare recursivă a parametrilor

3.1. Prezentare unitară

Metodele menționate în cadrul secțiunii precedente pot fi descrise în mod unitar de următorul algoritm:

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\,\varepsilon(t) \tag{3.1}$$

$$K(t)=\frac{P(t-1)\,z(t)}{1+\varphi(t)^T P(t-1)\,z(t)} \tag{3.2}$$

$$P(t)=P(t-1)-\frac{P(t-1)\,z(t)\,\varphi(t)^T P(t-1)}{1+\varphi(t)^T P(t-1)\,z(t)} \tag{3.3}$$

unde $\hat{\theta}$ reprezintă vectorul parametrilor estimați, utilizînd datele u (1), y (1), ..., u (t), y (t), fiind de forma:

Pentru RCMMP şi RVI :

$$\hat{\theta}=[\hat{a}_1, \ldots, \hat{a}_{n_a}, \hat{b}_1, \ldots, \hat{b}_{n_b}]^T \quad (3.4a)$$

şi pentru RCMMPG, RCMMPE, RVM :

$$\hat{\theta}=[\hat{a}_1, \ldots, \hat{a}_{n_a}, \hat{b}_1, \ldots, \hat{b}_{n_b}, \hat{c}_1, \ldots, \hat{c}_{n_c}]^T$$

(parametrii c_i au semnificaţii diferite pentru RCMMPG şi respectiv RCMMPE şi RVM) ;

$\varepsilon(t)$ reprezintă o estimare a erorii de predicţie pe un pas.

Parametrii sistemului (2.5) pot fi grupaţi într-un vector θ_0 al valorilor reale ale parametrilor, de forma :

Pentru RCMMP şi RVI

$$\theta_0=[a_1, \ldots, a_{n_a}, b_1, \ldots, b_{n_b}]^T \quad (3.4b)$$

şi pentru RCMMPG, RCMMPE, RVM

$$\theta_0=[a_1, \ldots, a_{n_a}, b_1, \ldots, b_{n_b}, c_1, \ldots, c_{n_c}]^T$$

cu aceeaşi observaţie, ca în cazul vectorului $\hat{\theta}$, pentru parametrii c_i.

Metodele în discuţie vor fi obţinute, sub forma unor cazuri particulare ale relaţiilor (3.1)—(3.3), prin specificarea variabilelor $\varphi(t)$ şi $z(t)$.

Metoda RCMMP. Algoritmul se obţine din relaţiile (3.1)—(3.3) cu următoarea alegere a vectorilor $\varphi(t)$, $z(t)$:

$$\varphi(t)=[-y(t-1), \ldots, -y(t-n_a), u(t-1), \ldots, u(t-n_b)]^T \quad (3.5)$$

$$z(t)=\varphi(t) \quad (3.6)$$

şi cu eroarea de predicţie calculată astfel :

$$\varepsilon(t)=y(t)-\varphi^T(t)\,\hat{\theta}(t-1) \quad (3.7)$$

Metoda RVI. Algoritmul de calcul se obţine prin alegerea vectorului $\varphi(t)$ şi a erorii de predicţie conform relaţiilor (3.5) şi respectiv (3.7), iar a vectorului z, al variabilelor instrumentale, astfel încît :

$$\text{(i)} \lim_{N\to\infty} \frac{1}{N}\sum_{t=1}^{N} z(t)\,\varphi(t)^T=R, \text{ R matrice nesingulară} \quad (3.8a)$$

$$\text{(ii)} \lim_{N\to\infty} \frac{1}{N}\sum_{t=1}^{N} z(t)\,H(q^{-1})\,e(t)=0 \quad (3.8b)$$

Alegerea elementelor vectorului $z(t)$ se poate face în mai multe moduri. O posibilitate de alegere este următoarea :

$$z(t)=[-x(t-1), \ldots, -x(t-n_a) ; u(t-1), \ldots, u(t-n_b)]^T \quad (3.9)$$

unde x (t) se obţine prin filtrarea semnalului de intrare u (t), conform relaţiei :

$$x(t)=\frac{\bar{B}(q^{-1})}{\bar{A}(q^{-1})}u(t) \quad \text{cu} \tag{3.10}$$

$$\bar{A}(q^{-1})=1+\bar{a}_1q^{-1}+\ldots+\bar{a}_{n_a}q^{-n_a}$$

$$\bar{B}(q^{-1})=\bar{b}_1q^{-1}+\ldots+\bar{b}_{n_b}q^{-n_b}$$

relativ prime (Finigan şi Rowe, 1974 ; Söderstrom, 1974a). În special, în cazul semnalelor de intrare de tip excitaţie persistentă, este indicată alegerea $\bar{A}(q^{-1})\equiv A(q^{-1})$, $\bar{B}(q^{-1})\equiv B(q^{-1})$. În acest caz x (t) devine egal cu partea deterministă a semnalului de ieşire. Această alegere este optimală (Wong şi Polak, 1967 ; Young, 1976), totuşi nu este posibilă deoarece sînt necesare valorile reale ale parametrilor modelului.

În locul variabilei x (t) se propune (Wong şi Polak, 1967 ; Young, 1968 ; Rowe, 1970) utilizarea variabilei :

$$\hat{x}(t)=z(t)^T\hat{\theta}(t) \tag{3.11}$$

şi modificarea în consecinţă a vectorului z (t).

S-au sugerat, de asemenea, următoarele modificări ale relaţiei (3.11) :

(i) trecerea vectorului parametrilor estimaţi printr-un element de întîrziere :

$$\hat{x}(t)=z(t)^T\hat{\theta}(t-\tau) \quad (\tau \text{ întreg mic pozitiv}) \tag{3.12}$$

(ii) trecerea vectorului parametrilor estimaţi printr-un filtru trece jos :

$$\hat{x}(t)=z(t)^T\tilde{\theta}(t) \tag{3.13}$$

$$\tilde{\theta}(t)=(1-\gamma)\tilde{\theta}(t-1)+\gamma\hat{\theta}(t) \quad (\gamma \text{ uşor subunitar})$$

Aceste modificări nu influenţează proprietăţile de convergenţă ale metodei, discutate în această lucrare.

Frecvent se utilizează $\bar{A}(q^{-1})\equiv 1$, $\bar{B}(q^{-1})\equiv q^{-n_b}$, vectorul variabilelor instrumentale conţinînd numai valori ale mărimii de intrare :

$$z(t)=[u(t-1), \ldots, u(t-n_a-n_b)]^T \tag{3.14}$$

Aşa cum s-a menţionat anterior metoda RVI nu determină un model al zgomotului. Din acest motiv s-a propus (Young, 1974) combinarea metodei RVI cu RCMMPE. Metoda sugerată determină, în prima etapă, utilizînd RVI, parametrii procesului $\hat{A}(q^{-1})$, $\hat{B}(q^{-1})$; în cea de-a două etapă se calculează pe baza valorilor parametrilor estimaţi, reziduurile modelului, care se utilizează ca date de intrare pentru metoda RCMMPE. Analiza proprietăţilor de convergenţă a metodei combinate se rezumă la analiza separată a proprietăţilor de convergenţă ale celor două metode.

Ulterior Young (1976) a propus o nouă schemă de estimare pentru metoda RVI, care permite şi calculul modelului zgomotului. În acest caz, estimarea dinamicii zgomotului este întreţesută cu estimarea dinamicii procesului (para-

metrii $\hat{A}(q^{-1})$, $\hat{B}(q^{-1})$). Metoda propusă este asemănătoare cu metoda RVM, dar nu este echivalentă cu aceasta.

În literatură sînt prezentate şi alte metode de identificare recursive similare metodei RVI. Una dintre ele o reprezintă metoda principiului informaţiei apriorice (tally principle) (Peterka-Haluskova, 1970).

Metoda RCMMPG. Varianta recursivă a metodei a fost sugerată de Hastings-James-Sage şi a fost inspirată de algoritmul off-line al lui Clarke (1967). Algoritmul include două estimatoare de tipul RCMMP cuplate prin faza de filtrare. Unul din estimatoare este utilizat pentru estimarea coeficienţilor $\hat{A}$ şi $\hat{B}$, în timp ce celălalt se utilizează pentru estimarea coeficienţilor $\hat{C}(q^{-1})$. Detalii asupra acestei metode sînt date de Hastings-James şi Sage (1969). Algoritmii similari au fost sugeraţi ulterior de Sen şi Sinha (1975) şi Gertler şi Bányász (1974).

Metoda RCMMPE. În acest caz

$$z(t)=\varphi(t)=[-y(t-1)\ldots-y(t-n_a),\ u(t-1),\ldots,u(t-n_b),\ \varepsilon(t-1),\ldots,\varepsilon(t-n_c)]^T \tag{3.15}$$

cu

$$\varepsilon(t)=y(t)-\varphi(t)^T\hat{\theta}(t-1) \tag{3.16}$$

Algoritmul este adesea utilizat pentru modelarea seriilor de timp (modele fără parametrii b_i), iar uneori pentru modelarea proceselor fără parametrii a_i. Pot fi utilizate şi alte structuri de modele, ca de exemplu cea propusă de Talmon şi van den Boom (1973):

$$\hat{A}(q^{-1})\,y(t)=\hat{B}(q^{-1})\,u(t)+\frac{\hat{C}(q^{-1})}{\hat{D}(q^{-1})}\,\varepsilon(t) \tag{3.17}$$

Metoda RVM. Metoda off-line a verosimilităţii maxime a fost dezvoltată de Aström şi Bohlin (1965), pentru modele descrise de relaţiile (2.1) şi (2.4). Metoda VM off-line corespunde minimizării funcţiei $\sum_{t=1}^{N}\varepsilon^2(t)$, unde $\varepsilon(t)$ este definit de (2.1). O versiune recursivă (aproximativă) a acestei metode este dată de Söderstrom (1973). Algoritmi similari sînt prezentaţi de Furht (1973) şi de Gertler şi Bányász (1974). Metoda RCMMPE şi cea propusă de Young (1976) pot fi interpretate ca variante aproximative ale metodei RVM.

Metoda RVM este descrisă de ecuaţiile (3.1)—(3.3) cu

$$z(t)=\varphi(t)=\left[-\frac{1}{\hat{C}(q^{-1})}y(t-1),\ldots,-\frac{1}{\hat{C}(q^{-1})}y(t-n_a);\ \frac{1}{\hat{C}(q^{-1})}u(t-1),\ldots,\frac{1}{\hat{C}(q^{-1})}u(t-n_b);\ \frac{1}{\hat{C}(q^{-1})}\varepsilon(t-1),\ldots,\frac{1}{\hat{C}(q^{-1})}\varepsilon(t-n_c)\right]^T \tag{3.18}$$

Eroarea de predicţie $\varepsilon(t)$ se determină cu relaţia:

$$\hat{C}(q^{-1})\,\varepsilon(t)=\hat{A}(q^{-1})\,y(t)-\hat{B}(q^{-1})\,u(t) \tag{3.19}$$

Pentru a obţine eroarea de predicţie exactă, ecuaţia (3.19) trebuie rezolvată, începînd cu $t=0$, pentru fiecare nou set de măsurări. Aceasta necesită un timp de calcul considerabil, fapt pentru care, în determinarea erorii de predicţie $\varepsilon(t)$, se fac aproximări convenabile. O astfel de soluţie constă în determinarea erorii de predicţie $\varepsilon(t)$ cu relaţia:

$$\varepsilon(t)=y(t)-[-y(t-1),\ldots,-y(t-n_a);\ u(t-1),\ldots,u(t-n_b);\ \varepsilon(t-1),$$

$$\ldots,\varepsilon(t-n_c)]\,\hat{\theta}(t-1) \qquad (3.20)$$

Ecuaţia este iterată o singură dată pentru fiecare set de măsurări.

Pe durata efectuării calculelor polinomul $\hat{C}(z)$ poate deveni instabil, ceea ce va conduce la valori mari pentru $\varepsilon(t)$ şi $\varphi(t)$, şi deci la neconvergenţa estimaţiilor parametrilor modelului. Depăşirea acestei dificultăţi se poate face prin reducerea termenului de corecţie $K(t)\,\varepsilon(t)$ în (3.1) astfel încît polinomul $\hat{C}(z)$, determinat de $\hat{\theta}(t)$, să fie stabil. Această modificare poate fi de asemenea avantajoasă şi în cazul metodei RCMMPE.

Pentru calculul elementelor vectorilor $z(t)$ şi $\varphi(t)$ în (3.18) se notează:

$$y^F(t)=\frac{1}{\hat{C}(q^{-1})}\,y(t)$$

rezultînd

$$y^F(t)=y(t)-\sum_{i=1}^{n_c}\hat{c}_i y^F(t-i) \qquad (3.21)$$

Această ecuaţie poate fi utilizată pentru generarea, aproximativ simplu, a întreg vectorului $\varphi(t)$. În calculul mărimii $y^F(t)$, în relaţia (3.21) se utilizează $\hat{\theta}(t-1)$. Variante aproximative de calcul al erorii de predicţie $\varepsilon(t)$ şi a vectorului $\varphi(t)$ sînt prezentate de Söderström (1973).

O comparaţie a metodelor RCMMPE şi RVM scoate în evidenţă multe similarităţi între cele două metode. Diferenţa constă în faptul că în cazul metodei RVM, în vectorul $\varphi(t)$, se utilizează date filtrate. Utilizarea filtrului $1/\hat{C}(q^{-1})$ derivă din relaţia $\varphi(t)^T=d\varepsilon(t)/d\theta$, care se dovedeşte foarte importantă pentru proprietăţile de convergenţă ale algoritmului.

Metoda poate fi uşor aplicată şi pentru alte structuri ale modelului. În această situaţie, algoritmul de bază rămîne neschimbat, dar trebuie utilizate alte relaţii de calcul pentru mărimile $\varepsilon(t)$ şi $\varphi(t)$.

Versiuni ale metodei aproximării stochastice. Algoritmii prezentaţi anterior se simplifică considerabil prin substituirea matricei $P(t)$ cu un scalar $p(t)$, de exemplu c/t sau $1/\mathrm{tr}\,P(t)^{-1}$, rezultînd algoritmi de forma:

$$\hat{\theta}(t)=\hat{\theta}(t-1)+p(t)\,z(t)\,\varepsilon(t) \qquad (3.22)$$

In această situaţie se obţine o reducere considerabilă a efortului de calcul, dar convergenţa algoritmilor rezultaţi va fi mai redusă decît în cazul algoritmilor originali.

Algoritmii de tipul (3.22) nu vor fi consideraţi explicit în cadrul acestei lucrări. Rezultatele analizei acestor algoritmi sînt similare rezultatelor obţinute pentru algoritmii originali.

3.2. Modificări ale algoritmilor

Versiuni în timp real. În multe situaţii este important ca valorile parametrilor modelului să fie estimate în timp real. În general, în literatură, sînt prezentate două soluţii de obţinere a versiunilor în timp real, pentru algoritmii de estimare recursivă descrişi de relaţiile (3.1) — (3.3).

O primă soluţie (Wieslander, 1969) constă în introducerea unui factor de ponderare λ a erorii de predicţie $\varepsilon(t)$ în expresia criteriului minimizat, rezultînd în final următoarele relaţii de calcul pentru vectorul parametrilor estimaţi :

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\cdot\varepsilon(t) \tag{3.23}$$

$$K(t)=\frac{P(t-1)\,z(t)}{\lambda+\varphi(t)^T P(t-1)\,z(t)} \tag{3.24}$$

$$P(t)=\left[P(t-1)-\frac{P(t-1)\,z(t)\,\varphi(t)^T P(t-1)}{\lambda+\varphi(t)^T P(t-1)\,z(t)}\right]/\lambda \tag{3.25}$$

Soluţia propusă realizează o reducere a influenţei reziduurilor „mai vechi" asupra valorii parametrilor estimaţi (pentru RCMMP criteriul de minimizat devine $\sum_{s=1}^{t} \lambda^{t-s}\varepsilon(s)^2$), prin alegerea factorului de ponderare λ subunitar.

O a doua soluţie (Bohlin, 1968) face uz de interpretarea de filtru Kalman, dată metodei RCMMP. Includerea zgomotului procesului în model conduce la următorul algoritm :

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\,\varepsilon(t) \tag{3.26}$$

$$K(t)=\frac{P(t-1)\,z(t)}{1+\varphi(t)^T P(t-1)\,z(t)} \tag{3.27}$$

$$P(t)=P(t-1)-\frac{P(t-1)\,z(t)\,\varphi(t)^T P(t-1)}{1+\varphi(t)^T P(t-1)\,z(t)}+R1 \tag{3.28}$$

unde R1 este o matrice (semi)pozitiv definită.

Cele două extensii au proprietatea că $P(t)$ (deci şi factorul de amplificare $K(t)$) nu tinde la zero cînd $t\to\infty$.

Îmbunătăţirea vitezei de convergenţă. Creşterea vitezei de convergenţă a algoritmilor de estimare recursivă se poate obţine prin utilizarea următoarei versiuni a algoritmului general (Furth, 1973 ; Talman, van den Boom, 1973) :

$$\hat{\theta}(t)=\hat{\theta}(t-1)+K(t)\,\varepsilon(t) \tag{3.29}$$

$$K(t)=\frac{P(t-1)\,z(t)}{\lambda(t)+\varphi(t)^T P(t-1)\,z(t)} \tag{3.30}$$

$$P(t)=\left[P(t-1)-\frac{P(t-1)\,z(t)\,\varphi(t)^T P(t-1)}{\lambda(t)+\varphi(t)^T P(t-1)\,z(t)}\right]/\lambda(t) \qquad (3.31)$$

cu $\lambda(t)$ generat de ecuaţia cu diferenţe de ordinul întîi:

$$\lambda(t)=\lambda_0\lambda(t-1)+(1-\lambda_0),\ \lambda_0 \text{ şi } \lambda(0) \text{ uşor subunitari} \qquad (3.32)$$

Ecuaţia (3.32) indică faptul că $\lambda(t)$ tinde exponenţial la 1 cînd t tinde la infinit.

Algoritmul propus apare ca o generalizare a algoritmului de estimare în timp real (3.23)—(3.25). Dependenţa de timp a factorului de ponderare $\lambda(t)$ va avea ca efect reducerea influenţei primelor valori estimate ale parametrilor modelului pe măsură ce diferenţa dintre valorile parametrilor estimaţi şi θ_0 se reduce. Factorul de ponderare, spre deosebire de cel din cazul versiunii în timp real a algoritmului, tinde la 1 odată cu creşterea timpului, ceea ce va conduce la obţinerea unei precizii mai bune a valorilor parametrilor estimaţi şi în consecinţă la îmbunătăţirea convergenţei.

În cazul metodei RCMMP, algoritmul de estimare realizează minimul următoarei funcţii obiectiv:

$$\sum_{k=1}^{t}\left[\prod_{i=k}^{t}\lambda(i)\right]\varepsilon(k)^2 \qquad (3.33)$$

Se observă clar profilul de „uitare" al datelor vechi, comparativ cu „uitarea" exponenţială din cazul versiunii în timp real a algoritmului. Relaţia (3.32) asigură următorul coeficient de uitare:

$$\lambda(t)=\lambda_0^t\lambda(0)+(1-\lambda_0^t)=1-\lambda_0^t(1-\lambda(0))$$

Algoritmul recursiv general poate fi rescris sub o altă formă, utilă consideraţiilor ce se vor face în cadrul secţiunilor viitoare.

Pentru o secvenţă dată $\{\lambda(t)\}$, care poate să nu verifice relaţia (3.32), se defineşte în mod recursiv următoarea secvenţă:

$$\gamma(t)=\frac{\gamma(t-1)}{\lambda(t)+\gamma(t-1)};\quad \gamma(0)=1 \qquad (3.34)$$

Atunci
$$\lambda(t)=\frac{\gamma(t-1)}{\gamma(t)}(1-\gamma(t)) \qquad (3.55)$$

Se introduc următoarele mărimi:

$$P^*(t)=\frac{1}{\gamma(t)}P(t) \qquad K^*(t)=\frac{1}{\gamma(t)}K(t)$$

Cu aceste notaţii ecuaţiile (3.29)—(3.31) devin:

$$\hat\theta(t)=\hat\theta(t-1)+\gamma(t)\,K^*(t)\,\varepsilon(t) \qquad (3.36)$$

$$K^*(t)=\frac{P^*(t-1)\,z(t)}{1+\gamma(t)\,[\varphi(t)^T P^*(t-1)\,z(t)-1]} \qquad (3.37)$$

$$P^*(t)^{-1}=P^*(t-1)^{-1}+\gamma(t)\,[z(t)\,\varphi(t)^T-P^*(t)^{-1}] \qquad (3.38)$$

sau
$$P^*(t)=\frac{1}{1-\gamma(t)}P^*(t-1)-\frac{\gamma(t)}{1-\gamma(t)}\,\frac{P^*(t-1)\,z(t)\,\varphi(t)^T P^*(t-1)}{1+\gamma(t)\,[\varphi(t)^T P^*(t-1)\,z(t)-1]}$$

Această descriere poate fi utilă, deoarece P^* şi K^* nu tind la 0 cînd $t \to \infty$ ci către limite ce pot avea interpretare fizică.

Algoritmi de tipul celui prezentat sînt studiaţi de Ljung (1974). În aceeaşi lucrare este discutată şi alegerea secvenţei $\gamma(t)$, în particular fiind prezentate motivele pentru care secvenţele lent descrescătoare $\{\gamma(t)\}$ (corespunzătoare factorului $\lambda(t)<1$) sînt mai indicate decît $\gamma(t)=1/t$.

4. Proprietăţi de convergenţă

Analiza directă a proprietăţilor de convergenţă a algoritmilor de estimare recursivă este dificil de realizat. Unul din motive îl constituie faptul că algoritmul (3.1)—(3.3) reprezintă o ecuaţie cu diferenţe stochastică, neliniară, variantă în timp. Mai mult, în general $\varepsilon(t)$ în (3.1) depinde implicit de toate valorile parametrilor estimaţi anterior $\widehat{\theta}(k)$, $k<t$. Toate acestea fac relativ dificilă tratarea analitică directă.

O soluţie posibilă de studiere a diferitelor metode de estimare recursivă o reprezintă fără îndoială simularea. Totuşi, adesea este destul de dificil să se elaboreze o concluzie asupra proprietăţilor algoritmilor studiaţi numai prin simulare. Rezultatele simulării sînt influenţate de mai mulţi factori ca, de exemplu, de realizarea proceselor aleatoare, de alegerea exemplelor etc.

În cadrul acestei secţiuni se face o sinteză a rezultatelor analitice obţinute în analiza algoritmilor recursivi, bazate pe tehnica generală de analiză a algoritmilor stochastici recursivi elaborată de Ljung (1977b).

Analiza se bazează pe asocierea algoritmului (3.1)—(3.3), unei ecuaţii diferenţiale deterministe şi invariante în timp (EDO), ale cărei proprietăţi reflectă proprietăţile algoritmului:

$$\frac{d\theta(\tau)}{d\tau}=R^{-1}(\tau)\, f(\theta(\tau))$$

$$\frac{dR(\tau)}{d\tau}=G(\theta(\tau))-R(\tau) \tag{4.1}$$

unde

$$f(\theta)=Ez(t;\theta)\,\varepsilon(t;\theta) \tag{4.2a}$$

$$G(\theta)=Ez(t;\theta)\,\varphi(t;\theta)^T \tag{4.2b}$$

E — operatorul de mediere.

În cadrul acestor relaţii $\varepsilon(t;\theta)$, $\varphi(t;\theta)$ şi $z(t;\theta)$ reprezintă procesele staţionare care s-ar obţine, dacă secvenţa parametrilor estimaţi $\{\widehat{\theta}(t)\}$ este înlocuită prin vectorul parametrilor constanţi θ.

Relaţiile dintre algoritmul (3.1)—(3.3), definind metodele de estimare recursivă, şi ecuaţia diferenţială (4.1) sînt următoarele (Ljung, 1977b):

— posibilele puncte de convergenţă ale algoritmului sînt numai punctele staţionare stabile ale ecuaţiei (4.1). Aceasta înseamnă, că dacă $\hat{\theta}(t) \rightarrow \theta^*$ cu probabilitate diferită de zero, atunci:

$$f(\theta^*)=0 \tag{4.3}$$

şi matricea

$$G^{-1}(\theta^*)\frac{df}{d\theta}(\theta)\Big|_{\theta=\theta^*} \tag{4.4}$$

are toate valorile proprii în semiplanul stîng;

— stabilitatea asimptotică globală a unei soluţii $(\theta^*, G(\theta^*))$ a ecuaţiei (4.1) implică $\hat{\theta}(t) \rightarrow \theta^*$ cu probabilitate 1 cînd $t \rightarrow \infty$, pentru algoritmul (3.1)—(3.3);

— traiectoriile ecuaţiei (4.1) pot fi interpretate ca traiectorii asimptotice ale algoritmului de estimare.

În continuare, vom prezenta o sinteză a rezultatelor analizei de convergenţă a metodelor de estimare recursivă, analiză bazată pe abordarea ce utilizează EDO.

În general se presupune că sistemul şi modelul sînt descrise de (2.5) şi respectiv (2.1). Interpretarea filtrelor $H(q^{-1})$ şi $\hat{H}(q^{-1})$ pentru diferite metode este dată în secţiunea 2. Din motive tehnice se consideră numai punctele pentru care $G(\theta)$ este nesingulară.

În cele ce urmează, pentru diferitele metode de estimare recursivă, se va da un răspuns următoarelor probleme:

(i) θ_0 este soluţia unică a ecuaţiei $f(\theta)=0$?

(ii) θ_0 este soluţia unică a ecuaţiei $\{f(\theta)=0$, şi $G^{-1}(\theta)f'(\theta)$ stabilă $\}$?

(iii) $\hat{\theta}(t)$ converge către θ_0?

în următoarele ipoteze:

— sistemul este identificabil, în sensul că acesta poate fi regăsit în structura modelului considerat;

— ordinele polinoamelor modelului coincid cu cele ale polinoamelor respective în descrierea sistemului (ordinul modelului este egal cu ordinul sistemului real);

— sînt analizate numai punctele pentru care $G(\theta)$ este nesingulară.

Motivele separării punctelor (i) şi (ii) sînt următoarele:

— ecuaţia $f(\theta)=0$ determină simultan şi proprietăţile de unicitate ale metodelor de estimare nerecursivă corespunzătoare;

— proprietăţile de unicitate ale soluţiei ecuaţiei $f(\theta)=0$ sînt esenţiale pentru demonstrarea convergenţei.

Rezultatele analizei convergentei sînt prezentate în tabelul 1.

Tabelul 1

Rezultate / Metoda	(i)	(ii)	(iii)
RCMMP	$\theta=\theta_0$ soluţia unică a ecuaţiei $f(\theta)=0$ (Söderström, Ljung, Gustavsson, 1974)	$G^{-1}(\theta_0)\,f'(\theta_0)$ stabilă (Söderström, Ljung, Gustavsson, 1974)	$\hat{\theta}(t) \rightarrow \theta_0$ c.p.1 (Söderström, Ljung, Gustavsson, 1974)
RVI	$\theta=\theta_0$ soluţia unică a ecuaţiei $f(\theta)=0$ (Procesul operează în buclă deschisă sau $H(q^{-1})\equiv 1$) (Söderström, Ljung, Gustavsson, 1974)	$G^{-1}(\theta_0)\,f'(\theta_0)$ stabilă (Procesul operează în buclă deschisă sau $H(q^{-1})\equiv 1$) (Söderström, Ljung, Gustavsson, 1974)	$\hat{\theta}(t) \rightarrow \theta_0$ c.p.1 pentru $z(t)$ generat de (3.9), (3.10); cazul (3.9), (3.11) rămîne deschis (Procesul operează în buclă deschisă sau $H(q^{-1})\equiv 1$) (Söderström, Ljung, Gustavsson, 1974)
RCMMPG	$\theta=\theta_0$ soluţie unică a ecuaţiei $f_i(\theta)=0$ ($i=1, 2$) pentru raportul semnal/zgomot foarte mare; în caz contrar există şi alte soluţii (Procesul operează în buclă deschisă) (Söderström, 1974b)	Sistemul liniar obţinut prin liniarizarea ecuaţiei (4.1) este asimptotic stabil dacă liniarizarea este făcută în jurul punctului θ_0 (valabil şi în cazul operării în buclă închisă a sistemului). Există cazuri unde de asemenea şi liniarizarea în jurul altor soluţii $f_i(\theta)=0$ ($i=1, 2$) furnizează un sistem asimptotic stabil. (Söderström, Ljung, Gustavsson, 1974)	$\hat{\theta}(t) \rightarrow \theta$ c.p.1. dacă reacţia nu este adaptivă şi $f_1(\theta)=0$ ($i=1, 2$) are soluţie unică $\theta=\theta_0$ (Söderström, Ljung, Gustavsson, 1974)
RCMMPE	$\theta=\theta_0$ soluţie unică a ecuaţiei $f(\theta)=0$ dacă este îndeplinită una din condiţiile: a) Sistemul este un proces ARMA $A(q^{-1})\,y(t)=B(q^{-1})\,e(t)$ (Söderström, Ljung, Gustavsson, 1974) b) Polinomul $C(q^{-1})\,e(t)$ (2.5), (2.8) este real	a) Pentru un proces ARMA, valorile proprii ale matricei $G^{-1}(\theta_0)\,f'(\theta_0)$ sînt -1 (ordin de multiplicitate n_c) şi $\frac{1}{C(\lambda_i^{-1})}$ $i=1, \ldots, n_a$ unde $\{\lambda_i\}$ sînt rădăcinile ecuaţiei $A(z)=0$ (Prin urmare există procese ARMA pentru care această matrice este instabilă) (Holst, 1977)	$\hat{\theta} \rightarrow \theta_0$ c.p.1 dacă $\mathrm{Re}\left[\frac{1}{C(e^{i\omega})} - \frac{1}{2}\right] > 0$ $-\pi \leqslant \omega \leqslant \pi$ (Ljung 1977 a)

Tabelul 1 (continuare)

Rezultate / Metoda	(i)	(ii)	(iii)
	pozitiv Re $C(e^{i\omega})>0$, $-\pi\leqslant\omega\leqslant\pi$ (Ljung 1977a)	b) Similar există procese a căror intrare este astfel încît $G^{-1}(\theta_0)\,f'(\theta_0)$ este instabilă (Ljung, Söderström, Gustavsson, 1975) c) Metoda propusă de Young (1976) are proprietăţi de convergenţă similare. Astfel pentru procese ARMA valorile proprii ale matricei $G^{-1}(\theta_0)\,f'(\theta_0)$ sînt $-1/C(\mu_i^{-1})$ $i=1,\ldots,n_c$ unde $\{\mu_i\}$ sînt rădăcinile ecuaţiei $C(z)=0$ şi $-1/C(\lambda_i^{-1})$ $i=1,\ldots,n_a$ unde $\{\lambda_i\}$ sînt rădăcinile ecuaţiei $A(z)=0$ (Holst 1977)	
RVM	$\theta=\theta_0$ soluţie unică a ecuaţiei $f(\theta)=0$ dacă sistemul este un proces ARMA (Aström, Söderström, 1974)	$G^{-1}(\theta_0)\,f'(\theta_0)$ întotdeauna asimptotic stabilă (Söderström, Ljung, Gustavsson, 1974) Rezultatele valabile şi pentru structuri generale ale modelului.	$\hat{\theta}\to\theta_0$ c.p.1. dacă reacţia este neadaptivă şi θ_0 este soluţia unică a ecuaţiei $f(\theta)=0$, $\hat{\theta}(t)$ converge întotdeauna către un maxim local al funcţiei de verosimilitate logaritmice. (Söderström, Ljung, Gustavsson, 1974) Rezultatele valabile şi pentru structuri generale ale modelului.

5. Aspecte de implementare

Cu toate mărimile definite în (3.29)—(3.32) sau (3.34)—(3.38), calculele ce trebuie efectuate pentru estimarea parametrilor modelului sînt unic definite. Totuşi, există mai multe variante, echivalente din punct de vedere algebric, pentru efectuarea aceloraşi calcule. Diferitele variante de organizare a calcule-

lor pot avea o influență esențială asupra proprietăților numerice ale algoritmului de estimare și, în special, asupra:
— timpului de calcul necesar efectuării unei iterații, conform relațiilor de estimare recursivă;
— necesarului de memorie calculator;
— preciziei și stabilității numerice a calculelor (propagării erorilor);
— efortului de programare pentru implementarea algoritmului.

Algoritmul de estimare constă în principal din trei părți. Prima parte, calculul erorii $\varepsilon(t)$, este banală. A doua parte, calculul vectorului de amplificare $K(t)$, este cea mai interesantă din punct de vedere al calculului numeric. Partea a treia a algoritmului, calculul vectorului parametrilor estimați $\theta(t)$, este directă, odată ce a fost determinat vectorul $K(t)$.

Aspectele de implementare ce se prezintă se vor referi, în special, la metodele de estimare ce utilizează algoritmii RCMMP, RCMMPE și RVM ($z(t)=\varphi(t)$).

5.1. Calculul vectorului de amplificare

Calculul vectorului de amplificare pentru algoritmii menționați anterior se efectuează cu relațiile:

$$P(t)=[P(t-1)-P(t-1)\varphi(t)S^{-1}(t)\varphi^{T}(t)P(t-1)]/\lambda(t) \quad (5.1a)$$

$$S(t)=\varphi^{T}(t)P(t-1)\varphi(t)+\lambda(t) \quad (5.1b)$$

$$K(t)=P(t-1)\varphi(t)S^{-1}(t) \quad (5.1c)$$

sau

$$R(t)=R(t-1)+\gamma(t)[\varphi(t)\varphi^{T}(t)-R(t-1)] \quad (5.2a)$$

$$K(t)=\gamma(t)R^{-1}(t)\varphi(t) \quad (5.2b)$$

cînd se lucrează cu inversa matricii de covarianță a vectorului parametrilor:

$$R(t)=\gamma(t)P^{-1}(t) \quad (5.3)$$

Factorul de „uitare" $\lambda(t)$ este legat de secvența $\gamma(t)$ prin relația (3.35)

$$\lambda(t)=\frac{\gamma(t-1)}{\gamma(t)}[1-\gamma(t)] \quad (5.4)$$

Trecerea de la o reprezentare la alta se realizează utilizînd lema de inversare.

Expresiile de mai sus apar în toți algoritmii. Acestea nu sînt influențate de structura modelului, ci numai de modul în care $\varphi(t)$ depinde de datele de măsurare: $u(t)$, $y(t)$.

În cadrul acestei secțiuni sînt analizate cîteva variante de calcul al vectorului $K(t)$. Varianta (5.2) nu este adecvată implementării, deoarece (5.2a) conține matricea $R(t)$ a cărei inversă este necesară în relația (5.2b). Dacă (5.2) se implementează în mod direct, la fiecare pas trebuie rezolvat un sistem de ecuații liniare, ceea nu nu apare necesar în cazul implementării relației (5.1). În continuare vor fi prezentate alte variante de calcul pentru vectorul de amplificare $K(t)$, care utilizează:

— *Tehnici de factorizare*. Acestea reprezintă alternative de calcul la utilizarea directă a ecuaţiei cu diferenţe (5.1). În acest caz matricea P (t) este exprimată ca un produs de factori matrice, efectuîndu-se actualizarea acestor factori, soluţie ce conferă proprietăţi numerice superioare metodei de estimare.

— *Algoritmi rapizi*. În cazul în care se urmăreşte numai calculul vectorului K (t), poate fi evitat calculul matricei P (t) sau R (t.) Această idee stă la baza algoritmilor rapizi, pentru care efortul de calcul şi necesarul de memorie se reduc semnificativ, în special pentru structuri de model cu mulţi parametri independenţi.

Algoritmii de estimare recursivă sînt strîns legaţi de filtrul Kalman. De fapt, multe din rezultatele ce se prezintă în cadrul acestei secţiuni au fost în primul rînd obţinute pentru filtrele Kalman. Deoarece (5.1), (5.2) sînt identice din punct de vedere algebric cu ecuaţiile factorului de amplificare şi de covarianţă ale filtrului Kalman, aceste rezultate pot şi vor fi aplicate în cadrul acestei secţiuni.

Utilizarea directă a relaţiei de recurenţă (5.1a) nu este indicată din punct de vedere numeric, fiind sensibilă la erori de rotunjire. Acest fenomen este analizat de Bierman (1977.) Proprietăţile numerice ale ecuaţiei (5.1a) sînt afectate în special pentru $\lambda(t)\equiv 1$. În acest caz, noua matrice P (t) este calculată prin scăderi succesive ale termenilor de corecţie. Dacă termenii de corecţie devin prea mari, datorită erorilor de rotunjire, pot apare probleme de natură numerică.

Ecuaţia (5.1a) poate fi rescrisă într-o formă care evident are proprietăţi numerice mai bune. Astfel (5.1a) este echivalentă cu :

$$P(t)=[I-K(t)\varphi(t)^T]\,P(t-1)\,[I-\varphi(t)K(t)^T]/\lambda(t) \qquad (5.5)$$

Această expresie este referită (Bierman 1977) ca „ecuaţia Kalman stabilizată" şi evită scăderile repetate. Cu toate acestea, relaţia de recurenţă (5.5) nu garantează stabilitatea numerică, deşi are proprietăţi numerice mai bune decît ecuaţia originală (5.1a). Totuşi, implementarea acesteia necesită un efort de calcul sporit.

Cînd dimensiunea matricei P este redusă ($d<10$), adesea nu apar probleme numerice. Aceste probleme sînt în mod obişnuit asociate unor valori mari ale dimensiunii matricei P (t) şi/sau matricilor P (t) prost condiţionate, totuşi aceasta nu reprezintă un motiv pentru ignorarea problemelor numerice.

Tehnici de factorizare pentru calculul vectorului de amplificare. În cazul utilizării unei tehnici de factorizare pentru calculul vectorului K (t), matricea P (t) din ecuaţia (5.1a) este descompusă într-un produs de factori matrice, fiecare din aceştia fiind actualizaţi la un nou moment de timp. Deoarece P (t) este pozitiv definită (neglijînd eventualele efecte ale erorilor de rotunjire), factorizarea poate fi de exemplu de tip Cholesky.

În cadrul acestei subsecţiuni, pentru început, se prezintă un algoritm de tip rădăcină pătrată, care realizează următoarea factorizare a matricei P (t) :

$$P(t)=Q(t)\,Q^T(t) \qquad (5.6)$$

unde Q (t) este o matrice nesingulară. În acest caz relaţia (5.1a) va fi înlocuită printr-una de calcul al matricei Q (t) din Q (t—1).

Este de asemenea posibil să se efectueze factorizarea (5.6) cu Q (t) matrice triunghiulară. (Bierman 1977), factorizarea fiind de tip Cholesky. În acest caz

algoritmul de actualizare a matricei Q (t) necesită mai multe calcule, dar este relativ simplu.

O altă metodă de factorizare frecvent utilizată este factorizarea U—D, care poate fi caracterizată drept o factorizare Cholesky normalizată. În acest caz :

$$P(t) = U(t)\,D(t)\,(U(t))^T \tag{5.7}$$

unde U (t) este o matrice triunghiulară cu elementele de pe diagonală unitate, iar D (t) este o matrice diagonală.

Utilizarea tehnicilor de factorizare, în calculul matricei P (t), asigură faptul că aceasta rămîne tot timpul pozitiv definită. De asemenea, experimentele intensive în cazul filtrului Kalman au scos în evidenţă faptul că metodele de factorizare menţionate au o bună stabilitate numerică, erorile de rotunjire neafectînd în mod semnificativ soluţia (Bierman, 1977 ; Thornton şi Bierman, 1978).

În continuare sînt prezentaţi algoritmul de tip rădăcină pătrată, în variantele propuse de Potter (1963) şi Bierman (1977), şi algoritmul de tip factorizare U—D propus de Bierman (1977).

A l g o r i t m u l r ă d ă c i n ă p ă t r a t ă (P o t t e r, 1963)

Algoritmul se bazează pe factorizarea (5.6), actualizarea matricii Q (t) făcîndu-se astfel :

$$\begin{aligned}
&1.\ f(t) := Q^T(t-1)\,\varphi(t)\\
&2.\ \pi(t) := \lambda(t) + f^T(t)\,f(t)\\
&3.\ \alpha(t) := 1/(\pi(t) + (\pi(t)\,\lambda(t))^{1/2}\\
&4.\ \bar{K}(t) := Q(t-1)\,f(t)\\
&5.\ Q(t) := [Q(t-1) - \alpha(t)\,\bar{K}(t)\,f^T(t)]/\lambda(t)^{1/2}
\end{aligned} \tag{5.8}$$

Vectorul $\bar{K}(t)$ reprezintă forma normalizată a vectorului de amplificare K (t) :

$$K(t) = \bar{K}(t)/\pi(t)$$

Actualizarea parametrilor modelului $\hat{\theta}(t)$ nu necesită calculul explicit al vectorului K (t) :

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \bar{K}(t)\,[\varepsilon(t)/\pi(t)] \tag{5.9}$$

În cazul estimării parametrilor în timp real, conform celor menţionate în cadrul secţiunii 3.2, se pot utiliza următoarele ecuaţii, în locul relaţiilor (5.1a, 5.1b)

$$\tilde{P}(t) = P(t-1) - \frac{P(t-1)\,\varphi(t)\,\varphi^T(t)\,P(t-1)}{1 + \varphi(t)\,P(t-1)\,\varphi(t)} \tag{5.10a}$$

$$P(t) = \tilde{P}(t) + R_1(t) \tag{5.10b}$$

Algoritmul (5.8)) poate fi utilizat, de asemenea, pentru determinarea matricei $\tilde{P}(t) = \tilde{Q}(t)\,\tilde{Q}^T(t)$ din $P(t-1) = Q(t-1)\,Q^T(t-1)$. Singura problemă

de rezolvat este de a determina matricea Q (t) folosind (5.10b). Una din soluţii se prezintă în continuare. Fie $R_1(t)$ factorizat astfel:

$$R_1(t)=V(t)\,V^T(t) \tag{5.11}$$

unde V (t) este o matrice rectangulară de rang complet.

În cele mai multe dintre cazuri, $R_1(t)$ este o matrice diagonală, în care unele elemente de pe diagonală sînt egale cu 0, factorul V (t) fiind uşor de determinat. În continuare se aplică transformări ortogonale matricei rectangulare $[\tilde{Q}(t)\,V(t)]$, problema constînd în a determina o matrice ortogonală T (t) şi o matrice triunghiulară Q (t) astfel încît:

$$[\tilde{Q}(t)\,V(t)]\,T(t)=[Q(t)\;0] \tag{5.12}$$

Atunci rezultă:

$$\tilde{P}(t)+R_1(t)=\tilde{Q}(t)\,\tilde{Q}^T(t)+V(t)\,V^T(t)=[\tilde{Q}(t)\,V(t)]\,T(t)\,T^T(t)\begin{bmatrix}\tilde{Q}^T(t)\\ V^T(t)\end{bmatrix}=$$

$$=[Q(t)\;0]\begin{bmatrix}Q^T(t)\\ 0\end{bmatrix}=P(t)$$

aşa cum se specifică prin (5.10b).

Matricile T (t) şi Q (t) în (5.12) pot fi determinate folosind, de exemplu, factorizarea QR sau procedura de ortogonalizare Gram-Schmidt. Astfel de factorizări apar în algebra liniară numerică pentru rezolvarea problemelor de valori proprii şi a celor mai mici pătrate (Stewart, 1973), programele aferente fiind incluse în pachetul LINPACK (Dongarra, Moler, Bunch şi Stewart, 1979).

Algoritmul rădăcină pătrată (Bierman, 1977)

Algoritmul factorizează matricea $P(t)=Q(t)\,Q(t)^T$, cu Q (t) matrice superior triunghiulară, incluzînd următoarele etape de calcul:

1. Se calculează: $f=Q^T(t-1)\,\varphi(t),\ \beta_0=\lambda(t)$
2. Pentru j=1, ..., n se execută paşii 3—5
3. Se calculează $\beta_j:=\beta_{j-1}+f_j^2$

$$\eta:=(\beta_{j-1}/\beta_j)^{1/2}$$

$$\nu:=f_j/(\eta\cdot\beta_j) \tag{5.13}$$

$$v_j:=f_j\cdot Q_{jj}(t-1)$$

$$Q_{jj}(t):=Q_{jj}(t-1)\cdot\eta$$

4. Pentru i=1, ..., j—1 se execută pasul 5 (pentru j=1 nu se execută pasul 5)
5. Se calculează: $Q_{ij}(t):=Q_{ij}(t-1)-v_i\cdot\nu$

$$v_i:=v_i+f_jQ_{ij}(t-1)$$

6. $\overline{K}:=v$

Vectorul $\overline{K}(t)$ reprezintă forma normalizată a vectorului de amplificare $K(t)$:

$$K(t)=\overline{K}(t)/\beta_n$$

Şi pentru acest algoritm se poate obţine o variantă adecvată estimării în timp real a parametrilor.

A l g o r i t m u l d e f a c t o r i z a r e U — D (B i e r m a n 1 9 7 7) Algoritmul utilizează factorizarea (5.7) şi include următoarele etape de calcul:

1. Se calculează: $f=U^T(t-1)\varphi(t)$, $g=D(t-1)f$, $\beta_0=\lambda(t)$
2. Pentru $j=1,\ldots,n$ se execută paşii 3—5
3. Se calculează $\beta_j:=\beta_{j-1}+f_j g_j$

$$D(t)_{jj}:=\beta_{j-1}D(t-1)_{jj}/\beta_j\lambda(t) \tag{5.14}$$

$$v_j:=g_j$$

$$\mu:=-f_j/\beta_{j-1}$$

4. Pentru $i=1,\ldots,j-1$ se execută pasul 5 (pentru $j=1$ nu se execută pasul 5)
5. Se calculează: $U(t)_{ij}:=U(t-1)_{ij}+v_i\cdot\mu$

$$v_i:=v_i+U(t-1)_{ij}v_j$$

6. $\overline{K}(t):=v$

Scalarul β_n obţinut după cel de al n-lea ciclu de execuţie a paşilor 3—5 reprezintă dispersia inovaţiei:

$$\beta_n=\lambda(t)+\varphi^T(t)P(t-1)\varphi(t)$$

Ca şi în cazul precedent, se obţine factorul de amplificare normalizat $\overline{K}(t)$ (pasul 6 după n cicluri de execuţie a paşilor 3—5).

De asemenea, şi în acest caz se poate obţine o variantă a algoritmului, adecvată estimării în timp real a parametrilor. Algoritmul ce se prezintă a fost propus de Thornton şi Bierman (1978) şi utilizează ortogonalizarea Gram-Schmidt.

Presupunem că factorizarea matricei $R_1(t)$ este disponibilă. Se construieşte o matrice W ale cărei linii sînt vectorii $W_1^T \ldots W_t^T$, astfel încît:

$$W=\begin{bmatrix} W_1^T \\ \vdots \\ W_n^T \end{bmatrix}=[\tilde{U}(t)\quad V(t)] \text{ şi se alege } D=\begin{bmatrix} \hat{D}(t) & 0 \\ 0 & I \end{bmatrix}$$

matricea $\tilde{P}(t)$ din (5.10) fiind factorizată astfel:

$$\tilde{P}(t)=\tilde{U}(t)\tilde{D}(t)\tilde{U}^T(t)$$

Atunci factorizarea U—D a matricei P (t), conform relației (5.10b), se poate scrie:

$$P(t)=U(t)\,D(t)\,U^T(t)=\tilde{U}(t)\,\tilde{D}(t)\,\tilde{U}^T(t)+V(t)\,V^T(t)=WDW^T$$

Vom încerca acum să determinăm matricile U (t) și $\tilde{W}$ astfel încît U (t) să fie matrice superior triunghiulară cu elemente unitate pe diagonală și

$$W=U(t)\,\tilde{W}$$

$$\tilde{W}D\tilde{W}^T=D(t)$$

Problema poate fi rezolvată aplicînd procedura de ortogonalizare Gram-Schmidt vectorilor $W_1 \ldots W_n$ folosind produsul scalar

$$<W_i,\ W_j>=W_i^T D W_j$$

Algoritmul final de actualizare în timp real al factorilor U—D pentru (5.10b) este următorul:

1. Se aleg $W_j^{(0)}:=W_j \quad j=1\ldots n$
2. Pentru $j=n, n-1, \ldots 2$ se execută pașii 3—5
3. Se calculează

$$D(t)_{jj}:=W_j^{T(n-j)}\,DW_j^{(n-j)}$$

4. Pentru $i=1, 2, \ldots, j-1$ se execută pasul 5 (5.15)
5. Se calculează

$$U(t)_{ij}:=W_i^{T(n-j)}\,DW_j^{(n-j)}/D(t)_{jj}$$

$$W_i^{(n-j+1)}=W_i^{(n-j)}-U(t)_{ij}W_j^{(n-j)}$$

6. Se calculează:

$$D(t)_{11}=W_1^{T(n-1)}\,DW_1^{(n-1)}$$

Complexitatea calculelor

O evaluare a numărului de operații aritmetice necesare pentru actualizarea matricei P (t) în cadrul algoritmilor de factorizare prezentați este dată în tabelul 2 (Bierman, 1977).

Tabelul 2

Metoda	Numărul de operații aritmetice			
	Adunări	Înmulțiri	Împărțiri	Extrageri rădăcină pătrată
Ecuația Kalman convențională	$1{,}5d^2+3{,}5d$	$1{,}5d^2+4{,}5d$	1	0
Ecuația Kalman stabilizată	$4{,}5d^2+5{,}5d$	$4d^2+7{,}5d$	1	0
Algoritmul de factorizare rădăcină pătrată (Potter)	$3d^2+3d$	$3d^2+4d$	2	1
Algoritmul de factorizare rădăcină pătrată (Bierman)	$1{,}5d^2+3{,}5d$	$2d^2+5d$	2	1
Algoritmul de factorizare U—D	$1{,}5d^2+1{,}5d$	$1{,}5d^2+5{,}5d$	d	0

Din acest tabel rezultă că algoritmii de factorizare U—D necesită, aproximativ, acelaşi efort de calcul ca ecuaţiile Kalman convenţionale. Actualizarea matricei de covariantă reprezintă numai o etapă în cadrul algoritmului de estimare, numărul operaţiilor necesare pentru calculul vectorului parametrilor fiind mult mai mare decît cel reprezentat în tabelul 2. Calculele necesită determinarea erorii de predicţie $\varepsilon(t)$ şi construcţia vectorului $\varphi(t)$, care depind în mare măsură de structura de model utilizată.

5.2. Algoritmi rapizi pentru calculul vectorului de amplificare

Algoritmii de calcul al vectorului de amplificare K (t) prezentaţi în secţiunea anterioară sînt valabili pentru orice structură a vectorului $\varphi(t)$. Algoritmii din cadrul acestei secţiuni utilizează o structură specifică a vectorului $\varphi(t)$, care se regăseşte în cadrul celor mai multe dintre tipurile de modele utilizate.

În cele ce urmează se defineşte această structură şi modul în care aceasta poate fi utilizată pentru calculul rapid al vectorului K (t). În acest caz, efortul de calcul, pe pas de estimare, devine proporţional cu d faţă de cel necesar în cazul convenţional, proporţional cu d^2 (d reprezintă dimensiunea vectorului parametrilor modelului).

Structura de deplasare (shift). În cele mai multe dintre cazuri $\{\varphi(t)\}$ are o structură particulară, în sensul că $\varphi(t)$ şi $\varphi(t+1)$ sînt dependente. Structura ce va fi presupusă în continuare, pentru $\varphi(t)$, este următoarea:

$$\varphi(t)=\begin{bmatrix} x(t-1) \\ \vdots \\ x(t-n) \end{bmatrix} \tag{5.16}$$

unde x (.) reprezintă un vector coloană cu α elemente. În consecinţă, dimensiunea vectorului $\varphi(t)$ este $\alpha n=d$.

Caracteristica cea mai importantă a structurii vectorului $\varphi(t)$ o reprezintă faptul că $\varphi(t+1)$ şi $\varphi(t)$ au un număr de elemente comune: $x(s)$; $t-n+1\leqslant s\leqslant t-1$.

Introducînd vectorul:

$$\varphi^*(t):=\begin{bmatrix} x(t) \\ \varphi(t) \end{bmatrix} \tag{5.17}$$

aceasta poate fi exprimat astfel:

$$\varphi^*(t)=\begin{bmatrix} x(t) \\ \varphi(t) \end{bmatrix}=\begin{bmatrix} \varphi(t+1) \\ x(t-n) \end{bmatrix} \tag{5.18}$$

Se poate, de asemenea, presupune o structură ceva mai generală astfel încît:

$$\begin{bmatrix} x(t) \\ \varphi(t) \end{bmatrix}=S_F\varphi^*(t)\,; \quad \begin{bmatrix} \varphi(t+1) \\ \tilde{x}(t-n) \end{bmatrix}=S_B\varphi^*(t) \tag{5.19}$$

unde S_F şi S_B sînt matrici de permutare, iar x (t) şi $\tilde{x}$ (t) nu sînt în mod necesar aceeaşi. Evident (5.18) este un caz special pentru (5.19). Totuşi, din considerente de simplitate a scrierii, în cele ce urmează, se va trata cazul (5.18).

Calculul rapid al vectorului de amplificare. În cadrul acestei secţiuni se prezintă algoritmul de calcul al vectorului de amplificare K (t) (5.2) (Ljung, Morf şi Falconer ; 1978) care utilizează structura de deplasare (5.16).

Presupunem următoarea condiţie iniţială pentru ecuaţia (5.2) :

$$R(0)=\delta I \text{ cu } \delta\geqslant 0$$

Introducînd

$$\bar{R}(t)=\frac{1}{\gamma(t)}R(t)$$

ecuaţia (5.2 a) devine :

$$\bar{R}(t)=\lambda(t)\bar{R}(t-1)+\varphi(t)\varphi^T(t) \tag{5.20}$$

cu λ (t) dat de (5.4).

În continuare se va presupune că :

$$\lambda(t)\equiv\lambda, \text{ pentru toate valorile timpului } t. \tag{5.21}$$

Prin urmare

$$\bar{R}(t)=\sum_{k=1}^{t}\lambda^{t-k}\varphi(k)\varphi^T(k)+\lambda^t\delta.I \tag{5.22}$$

În aceste condiţii vectorul de amplificare K (t) reprezintă soluţia ecuaţiei :

$$\bar{R}(t)K(t)=\varphi(t) \tag{5.23}$$

cu $\bar{R}$ (t) dat de (5.20)—(5.22).

Ideea de bază a algoritmului de calcul rapid este de a utiliza structura (5.16) sau (5.18) pentru rezolvarea ecuaţiei (5.23). Această structură permite stabilirea unei legături directe între K (t) şi K (t+1), aşa cum va rezulta în continuare.

Fie φ^* (t) definită de (5.17), şi :

$$\bar{R}^*(t)=\sum_{k=1}^{t}\lambda^{t-k}\varphi^*(k)[\varphi^*(k)]^T+\lambda^t\delta.I \tag{5.24}$$

$\bar{R}^*$ (t) este o matrice $(n+1)\alpha^*(n+1)\alpha$.

Proprietatea (5.18) implică următoarea relaţie :

$$\bar{R}^*(t)=\begin{bmatrix} * & * & * \\ * & \bar{R}(t) & \\ * & & \end{bmatrix}=\begin{bmatrix} & & * \\ & \bar{R}(t+1) & * \\ * & * & * \end{bmatrix} \tag{5.25}$$

unde prin „*“ au fost notate elementele ce nu prezintă interes în momentul de faţă, iar „lăţimea“ acestor elemente este α. Relaţia (5.25) este valabilă numai dacă x(t)=0 pentru $t\leqslant 0$. Altfel, efectul termenului $\varphi^T(1)\varphi^T(1)$ nu va fi luat în considerare în cadrul egalităţii a doua.

Relaţiile de definiţie a vectorilor K (t) şi K (t+1)

$$\bar{R}(t)\,K(t)=\varphi(t)$$

$$\bar{R}(t+1)\,K(t+1)=\varphi(t+1)$$

pot fi rescrise, utilizînd relaţia (5.25), sub forma:

$$\bar{R}^*(t)\cdot\begin{bmatrix}0\\K(t)\end{bmatrix}=\begin{bmatrix}*\\\varphi(t)\end{bmatrix} \tag{5.26}$$

$$\bar{R}^*(t)\cdot\begin{bmatrix}K(t+1)\\0\end{bmatrix}=\begin{bmatrix}\varphi(t+1)*\end{bmatrix} \tag{5.27}$$

În acest caz „*" reprezintă un vector α*1, a cărui expresie exactă nu este importantă în definirea vectorului K (t).

În scopul determinării relaţiei dintre K (t+1) şi K (t) se introduce o mărime intermediară K* (t) care se defineşte astfel

$$\bar{R}^*(t)\,K^*(t)=\varphi^*(t) \tag{5.28}$$

În continuare se va urmări stabilirea unor relaţii, între K (t), K* (t) şi respectiv K* (t), K (t+1).

În scopul stabilirii dependenţelor menţionate anterior sînt necesare instrumente cu care să se poată opera asupra elementelor „*" în (5.26)—(5.27). Aceste instrumente sînt matricile (nα*α): A (t) şi B (t) care satisfac relaţiile:

$$\bar{R}^*(t)\begin{bmatrix}I\\\hdashline A(t)\end{bmatrix}=\begin{bmatrix}R^e(t)\\\hdashline 0\end{bmatrix}\updownarrow\ \alpha\ \text{linii} \tag{5.29}$$

$$\bar{R}^*(t)\begin{bmatrix}B(t)\\\hdashline I\end{bmatrix}=\begin{bmatrix}0\\\hdashline R^r(t)\end{bmatrix}\updownarrow\ \alpha\ \text{linii} \tag{5.30}$$

$R^e(t)$ şi $R^r(t)$ sînt matrici α*α.

Formulele pentru actualizarea matricilor A (t), B (t) şi a vectorului K (t) (Ljung, Morf şi Falconer ; 1978) sînt sintetizate în următorul algoritm de estimare rapidă :

Algoritmul de calcul. Fie {x (t)} o secvenţă de vectori α*1, astfel încît x (t)=0 pentru t$\leq$0 şi fie :

$$\varphi(t)=\begin{bmatrix}x(t-1)\\\vdots\\x(t-n)\end{bmatrix}$$

Atunci vectorul de amplificare :

$$K(t)=\left[\sum_{k=1}^{t}\lambda^{t-k}\varphi(k)\,\varphi^T(k)+\lambda^t\delta\cdot I\right]^{-1}\varphi(t)$$

poate fi calculat recursiv astfel:

1. Se iniţializează

$$A(0)=0,\ B(0)=0,\ R^e(0)=\delta \cdot I,\ K(1)=0$$

2. Fiind date A (t—1), B (t—1), R^e (t—1), K (t) se actualizează astfel:

$$e(t) := x(t) + A^T(t-1)\,\varphi(t)$$

$$A(t) := A(t-1) - K(t)\,e^T(t)$$

$$\beta(t) := K^T(t)\,\varphi(t)$$

$$\bar{e}(t) := (1-\beta(t))\,e(t)$$

$$R^e(t) := \lambda R^e(t-1) + \bar{e}(t)\,e^T(t)$$

$$K^*(t) = \begin{bmatrix} [R^e(t)]^{-1}\bar{e}(t) \\ K(t) + A(t)\,[R^e(t)]^{-1}\bar{e}(t) \end{bmatrix} \tag{5.31}$$

Se partiţionează K* (t) astfel:

$$K^*(t) := \begin{bmatrix} M(t) \\ \cdots \\ \mu(t) \end{bmatrix} \begin{matrix} n\alpha \text{ linii} \\ \\ \alpha \text{ linii} \end{matrix}$$

$$r(t) := x(t-n) + B(t-1)^T \varphi(t+1)$$

$$B(t) := [B(t-1) - M(t)\,r^T(t)]\,[I - \mu(t)\,r^T(t)]^{-1}$$

$$K(t+1) := M(t) - B(t)\,\mu(t)$$

Complexitatea calculelor. Algoritmul (5.31) necesită memorarea, la momentul t—1, a următoarelor elemente:

$\varphi(t)$: matrice $n\alpha*1$

$K(t)$: matrice $n\alpha*1$

$A(t-1)$: matrice $n\alpha*\alpha$

$B(t-1)$: matrice $n\alpha*\alpha$

$R^e(t-1)$: matrice simetrică $\alpha*\alpha$

În total acestea utilizează $2n\alpha^2+2n\alpha+\frac{1}{2}(\alpha^2+\alpha)$, sau $2\alpha d+2d+\frac{1}{2}(\alpha^2+\alpha)$, $d=n\alpha$, locaţii memorie.

Algoritmii de calcul al vectorului de amplificare, utilizînd tehnici de factorizare, necesită memorarea matricei simetrice P (t) (sau a matricei triunghiulare U (t) şi a matricei diagonale D (t)) împreună cu vectorul φ (t). Aceasta înseamnă $\frac{1}{2}(n\alpha)^2+\frac{3}{2}n\alpha$ locaţii memorie, deci cu un ordin de mărime (în n) mai mare.

Numărul de operaţii necesar în cadrul algoritmului (5.31) este de aproximativ $n\alpha^3+6n\alpha^2+n\alpha+\frac{2}{3}\alpha^3+4\alpha^2+2\alpha$ înmulţiri şi $n\alpha^3+7n\alpha^2+2n\alpha+7/2\alpha^2+3/2\alpha$

adunări fiind cu un ordin de mărime mai mic (în n) decît cel necesar în cadrul algoritmilor de factorizare, conform tabelului 2.

Prin urmare, utilizarea algoritmului (5.31) este avantajoasă, în raport cu metodele convenţionale cînd n este mare : algoritmul este mai rapid şi necesită mai puţină memorie calculator.

Stabilitatea numerică. Proprietăţile algoritmului (5.31) legate de propagarea erorilor nu sînt încă, în întregime, înţelese şi investigate. Este posibil ca reducerea redundanţei să conducă la degradarea stabilităţii numerice a algoritmului. Cele mai multe din studiile publicate indică, totuşi, o comportare satisfăcătoare a algoritmului (Ljung, Morf şi Falconer, 1978 ; Robins şi Wellstead, 1979). Totuşi, ocazional, se observă creşteri exagerate ale valorii elementelor matricei $R^e(t)$ cînd $\lambda<1$.

Extensii ale algoritmului. Structura generală de deplasare (5.19) este tratată de Ljung, Morf şi Falconer (1978). Această referinţă tratează de asemenea şi metoda nesimetrică, a matricei instrumentale. Cazul în care $\varphi(t)$ este matrice pate fi tratat în acelaşi mod cu cazul vectorial, procedura de obţinere a algoritmului fiind aceeaşi.

5.3. Regularizarea algoritmilor

Conform celor prezentate în secţiunea 5.1 factorul de amplificare $K(t)$ se determină cu relaţiile :

$$R(t)=R(t-1)+\gamma(t)(\varphi(t)\varphi^T(t)-R(t-1)) \quad (5.32\text{ a})$$

$$K(t)=\gamma(t)R(t)^{-1}\varphi(t) \quad (5.32\text{ b})$$

Se poate întîmpla ca matricea $R(t)$, definită de (5.32 a), să devină singulară, sau aproape singulară. În general, această situaţie apare dacă modelul conţine prea mulţi parametri sau dacă semnalul de intrare nu este destul de general.

În implementarea algoritmilor de estimare recursivă în cadrul schemelor de comandă adaptivă, funcţionînd un timp îndelungat, sau în cadrul aplicaţiilor de supraveghere, este important să se aibă în vedere acest gen de probleme numerice, deoarece nu se poate garanta a priori că intrarea rămîne destul de generală.

În consecinţă, trebuie luate unele măsuri care să asigure că elementele matricei $R(t)$ rămîn mărginite, $|R^{-1}(t)|<C$. O astfel de ipoteză apare în analiza teoretică a algoritmilor (Ljung, Söderström, 1981) sub forma :

$$R(t)\geqslant\delta I\;;\quad\delta>0 \quad (5.33)$$

O tehnică standard de asigurare a condiţiei (5.3) este *regularizarea*. În cadrul acestei secţiuni se discută cîteva moduri de implementare a problemei de estimare recursivă cu asigurarea condiţiei (5.33).

Metoda Levenberg-Marquardt. Matricea $R(t)$ în (5.32 a) este prin construcţie semipozitiv definită. Deci, obţinerea unei relaţii de tipul (5.33) se poate

realiza simplu prin adăugarea termenului δI în relația (5.32 a), rezultînd următorul algoritm (regularizare Levenberg-Marquardt, Marquardt (1963)):

$$R(t)=R(t-1)+\gamma(t)(\varphi(t)\varphi^T(t)-R(t-1))+\delta I \quad (5.34)$$

Modificarea (5.34) este conceptual simplă. Numărul pozitiv poate fi ales destul de mic ($\delta=10^{-2}-10^{-4}$) comparativ cu valoarea elementelor vectorului $\varphi(t)$. Prin urmare direcțiile de căutare obținute din (5.34) vor fi puțin diferite de cele algoritmului inițial.

Există totuși un dezavantaj al implementării relației (5.34). Relația recursivă de calcul a matricei $P(t)=\gamma(t)R^{-1}(t)$, se obține aplicînd lema de inversare ecuației (5.32 a). Această relație va conține inversa unei matrici de dimensiune egală cu 1, rangul matricii $\varphi(t)\varphi^T(t)$, rang mult mai mic decît $d=\dim\theta$. În (5.34) această matrice este $\varphi(t)\varphi^T(t)+\delta I$ care are rang d. Deci (5.34) implică inversarea la fiecare pas a unei matrici $d*d$.

Un posibil remediu este următorul (Ljung, Söderström, 1981). În locul adăugării unei matrici δI, $d\times d$, se adaugă un element diagonal la fiecare pas. Fie $J_d(t)$ o matrice al cărui element diagonal t (mod d)+1 este 1, iar celelalte elemente sînt nule. Considerăm atunci algoritmul:

$$R(t)=R(t-1)+\gamma(t)(\varphi(t)\varphi^T(t)+d\delta J_d(t)-R(t-1)) \quad (5.35)$$

Acest algoritm este de fapt identic cu (5.34). După d pași se obține:

$$\sum_{t=k+1}^{k+d} d\delta J_d(t)=d\delta I$$

astfel încît se adaugă un multiplu al matricii unitate pentru a se asigura relația (5.33). Versiunea (5.35) se poate scrie:

$$R(t)=R(t-1)+\gamma(t)(\varphi^*\Lambda^*\varphi^{*T}(t)-R(t-1)) \quad (5.36\text{ a})$$

unde: $\varphi^*(t)$ este o matrice $d*2$

$$\varphi^*(t)=\begin{bmatrix} & 0 \\ & \vdots \\ & 0 \\ \varphi(t) & 1 \\ & 0 \\ & \vdots \\ & 0 \end{bmatrix} \rightarrow \text{poziția } t \text{ (mod d)}+1 \quad (5.36\text{ b})$$

și

$$\Lambda^*(t)=\begin{bmatrix}1 & 0\\ 0 & d\delta\end{bmatrix}$$

Lema de inversare aplicată relației (5.35) conduce la următorul algoritm:

$$P(t)=[P(t-1)-P(t-1)\varphi^*(t)S^*(t)^{-1}\varphi^*(t)^TP(t-1)]/\lambda(t) \quad (5.37\text{ a})$$

$$S^*(t)=\varphi^*(t)^TP(t-1)\varphi^*(t)+\lambda(t)\Lambda^*(t) \quad (5.37\text{ b})$$

unde:

$$P(t)=\gamma(t)R^{-1}(t)$$

şi $\lambda(t)$ dat de (5.4). S^* are dimensiunea 2 pentru adaptarea la modificarea Levenberg-Marquardt. În rest algoritmul pentru determinarea vectorului amplificării rămîne neschimbat.

Regularizarea algoritmului de factorizare U—D. Modificarea Levenberg-Marquardt (5.34) sau (5.37) adaugă la matricea $R(t)$ o matrice constantă pozitiv definită, fie că este necesar sau nu. Totuşi, pare mai natural a se efectua o astfel de modificare numai cînd există riscul ca valorile proprii ale matricei $R(t)$ să tindă la zero.

Un astfel de algoritm necesită, în mod aparent, un efort de calcul mai mare decît regularizarea directă.

În cazul algoritmului de factorizare U—D, informaţia suplimentară necesară noii versiuni de regularizare se obţine în mod automat.

Fie:

$$P(t)=U(t)\,D(t)\,U^T(t) \tag{5.38}$$

unde $U(t)$ este o matrice triunghiulară normalizată, iar $D(t)$ este o matrice diagonală.

Introducînd limite pentru elementele matricilor $D(t)$ şi $U(t)$ putem asigura, în consecinţă, că matricea $P(t)$ rămîne mărginită şi pozitiv semidefinită.

Din considerente practice este suficient, de fapt, să se introducă o limită numai pentru $D(t)$. Justificarea este următoarea:

$$\gamma(t)\,P^{-1}(t)=R(t)=\sum_{k=1}^{t}\beta(t,k)\,\varphi(k)\,\varphi^T(k)=\gamma(t)\,U^{-T}(t)\,D^{-1}(t)\,U^{-1}(t) \tag{5.39}$$

Presupunem că $\gamma(t)$ nu tinde la zero şi că vectorii $\varphi(t)$ nu tind la infinit. Atunci elementele matricei $U^{-1}(t)$ rămîn mărginite, ca şi elementele matricei $U(t)$ (deoarece $\det U(t)=1$). Prin urmare, este suficient să se introducă o limită numai pentru elementele lui $D(t)$. Regularizarea algoritmului U—D se obţine prin urmare prin modificarea pasului 3 din (5.14), astfel:

$$3.\; D(t)_{jj}:=\min\left(C,\ \beta_{j-1}\,D(t-1)_{jj}/\beta_j\lambda(t)\right) \tag{5.40}$$

unde C este un număr pozitiv ce mărgineşte elementele matricei $D(t)$. Acesta corespunde termenului $\gamma(t)/\delta$, cu δ ales conform discuţiei din secţiunea anterioară.

Regularizarea (5.40) este naturală şi uşor de implementat în cadrul algoritmului U—D.

5.4. Teste de stabilitate şi algoritmi de proiecţie

În demonstrarea proprietăţilor algoritmilor de estimare recursivă apare destul de des ipoteza că vectorul parametrilor estimaţi $\theta(t)$ aparţine, cel puţin infinit de des, submulţimii predictorilor stabili. Algoritmul de actualizare a parametrilor (3.29) se poate rescrie astfel:

$$\hat{\theta}(t)=[\hat{\theta}(t-1)+K(t)\,\varepsilon(t)]_{D_M}$$

realizînd proiecţia în submulţimea predictorilor stabili.

De asemenea, în cadrul unor situaţii practice este necesar ca $\hat{\theta}(t)$ să aparţină submulţimii predictorilor stabili. În consecinţă se impune ca algoritmii de estimare recursivă să conţină elemente de analiză a stabilităţii şi de obţinere a proiecţiei vectorului parametrilor în regiunea de stabilitate.

Algoritmii de proiecţie nu trebuie să fie foarte sofisticaţi. Cel mai frecvent facilitatea de proiecţie se implementează sub forma unei reduceri succesive a termenului de corecţie pentru vectorul parametrilor estimaţi. Aceasta poate fi realizată în felul următor:

1. Se alege un factor μ, $0 \leqslant \mu < 1$
2. Se calculează
$$\hat{\theta}(t) := \gamma(t)\, R^{-1}(t)\, \varphi(t)\, \varepsilon(t)$$
3. Se calculează
$$\hat{\theta}(t) := \hat{\theta}(t-1) + \tilde{\theta}(t)$$
4. Se testează dacă $\hat{\theta}(t) \in D_M$

   Dacă da se trece la 6, în caz contrar se trece la 5
5. Se calculează
$$\tilde{\theta}(t) := \mu \tilde{\theta}(t) \text{ şi se trece la } 3$$
6. Stop.

În etapa 4, testul dacă $\hat{\theta}(t) \in D_M$ constă în a examina dacă anumite polinoame au toate zerourile în interiorul cercului unitate. Aceasta este o problemă clasică în analiza sistemelor liniare discrete, existînd algoritmi de efectuare a acestor teste (Kucera, 1979).

În etapa 5, factorul μ realizează reducerea modificării vectorului parametrilor. Alegerea $\mu = 0{,}5$ pare a fi cea mai indicată.

6. Concluzii

Scopul acestei lucrări a fost de a prezenta într-un cadru unitar unele aspecte teoretice şi de implementare, specifice tehnicilor de estimare recursivă.

Algoritmii prezentaţi reprezintă ecuaţii cu diferenţe neliniare şi stochastice care fac ca analiza directă a convergenţei să fie extrem de dificilă. Analiza teoretică a algoritmului este posibilă prin studiul soluţiei unei ecuaţii diferenţiale ordinare asociate (EDO). Rezultatele analizei algoritmilor sintetizate în lucrare au fost obţinute în ipoteza că structura modelului este aceeaşi cu cea a sistemului considerat şi că matricea $G(\theta^*)$ este nesingulară.

În ceea ce priveşte aspectele de implementare a algoritmului de estimare recursivă general, a fost tratat în detaliu calculul vectorului de amplificare. Ca soluţie de implementare se sugerează utilizarea algoritmului de factorizare

U—D care încorporează simplu și facilitatea de regularizare. Aceste aspecte sînt importante, în special, pentru algoritmii care se utilizează în cadrul unor sisteme de conducere adaptivă a proceselor industriale.

BIBLIOGRAFIE

1. *Aström, K. J.* (1970). **Introduction to Stochastic Control Theory,** Academic Press, New York.
2. *Aström, K. J. și T. Bohlin* (1965). **Numerical identification of linear dynamic systems from normal operating records,** IFAC Symposium on Self-Adaptive Systems, Teddington.
3. *Aström, K. J. și P. Eykhoff* (1971). **System identification — A survey,** Automatica, 7, 123—162.
4. *Aström, K. J. și T. Söderstrom* (1974). **Uniqueness of the maximum likelihood estimates of the parameters of an ARMA model,** IEEE Trans. AC-19, 769—773.
5. *Bierman, G. J.* (1977). **Factorization algorithms for discrete sequential estimation,** Academic Press, New York.
6. *Bohlin, T.* (1968). **Real time estimation of time variable process characteristics,** Technical paper TP 18.190, IBM Nordic Lab., Lidingö.
7. *Clarke, D. W.* (1967). **Generalized least squares estimation of parameters of a dynamic model.** 1 st IFAC Symposium on Identification in Automatic. Control Systems, Prague.
8. *Dongarra, J. J., C. B. Moler, J. R. Bunch și G. W. Stewart* (1979). **LINPACK Users' Guide,** Siam, Philadelphia.
9. *Eykhoff, P.* (1974). **System Identification — Parameter and State Estimation,** J. Wiley & Sons, London.
10. *Finigan, B. și I. H. Rowe* (1974). **Strongly consistent estimation by the introduction of strong instrumental variable,** IEEE Trans. AC-19, 825—830.
11. *Furth, B. P.* (1973). **New estimator for the identification of dynamic processes,** IBK Report, Institut Boris Kidrič Vinča, Belgrad.
12. *Gertler, J. și Cs Bányász* (1974). **A recursive (on-line) maximum likelihood identification method,** IEEE Trans. AC-19, 816—820.
13. *Hannan, E. I.* (1976). **The convergence of some recursions,** Ann. Statist. 4, 1258—1270.
14. *Hasting-James, R. și M. W. Sage* (1969). **Recursive generalized least squares procedure for on—line identification of process parameters,** IEE Proc. 116, 2057—2062.
15. *Holst, J* (1977). **Adaptive prediction and recursive estimation,** Report 1013, Dept. of Automatic Control, Lund Inst. of Techl., Lund.
16. *Kucera V.* (1979). **Discrete Linear Control,** John Wiley and Sons, Chichester.
17. *Landau, I. D.* (1974). **A survey of model reference adaptive techniques : Theory and applications,** Automatica, 10, 353—379.
18. *Ljung, L.* (1974). **Convergence of recursive stochastic algorithms,** Report 7403, Dept. of Automatic Control, Lund Inst. of Tech., Lund.
19. *Ljung L., T. Soderstrom și I. Gustavsson* (1975). **Counterexamples to general convergence of a commonly used identification method,** IEEE Trans. AC-20, 643—652.
20. *Ljung, L.* (1977a). **On positive real functions and the convergence of some recursive schemes,** IEEE Trans. AC-22, 539—551.
21. *Ljung, L.* (1977b). **Analysis of recursive stochastic algorithms,** IEEE Trans, AC-22, 551—575.
22. *Ljung, L., M. Morf și D. Falconer* (1978). **Fast calculation of gain matrices for recursive estimation schemes,** Int. J. Control, 27, 1—18.
23. *Ljung, L., și T. Söderstrom* (1981). **Theory and Practice of Recursive Identification,** în curs de apariție la MIT Press.
24. *Marquardt, D.W.* (1963). **An algorithm for least-squares estimation of non-linear parameters** Journal SIAM, 11, 431—441.
25. *Mayne, D.Q.* (1967). **A method for estimating discrete time transfer functions.** In Advances in Computer Control, Second UKAC Control Convention, University of Bristol.
26. *Narendra, K.S. și P. Kudva* (1974). **Stable adaptive schemes for system identification control — Part. II,** IEEE Trans. on Systems, Man and Cybernetics, SMC-4, 552—560.
27. *Panuska, V.* (1968). **A stochastic approximation method for identification of linear systems using adaptive filtering,** Proc. JACC.

28. *Panuska, V.* (1969). **An adaptive recursive least squares identification algorithm.** Proc. IEEE Symposium on Adaptive Processes, Decision and Control.
29. *Parks, P.C.* (1966). **Lyapunov redesign of model reference adaptive control systems.** IEEE Trans. AC-11, 362—367.
30. *Peterka V.* şi *A. Haluskova* (1970). **Tally estimate of Aström model for stochastic systems,** Proc. 2 nd IFAC Symposium on Identification and Process Parameter Estimation, Prague.
31. *Popescu, Th.* (1980). **Metode de identificare recursive,** Referat de doctorat, Fac. de Automatică, Institutul Politehnic Bucureşti.
32. *Popescu, Th.* (1981). **Conducerea adaptivă cu calculator a sistemelor continue,** Referat de doctorat, Fac. de Automatică, Institutul Politehnic Bucureşti.
33. *Potter, J.E.* (1963) **New statistical formulas.** Memo 40, Instrumentation Laboratory, Massachusetts Institute of Technology, Cambridge, Mass.
34. *Robins, A.J.* şi *P.E. Wellstead* (1981). **Recursive system identification using fast algorithms,** Int. J. Control, 33, 455—480.
35. *Rowe, I.H.* (1970). **A bootstrap method for the statistical estimation of model parameters,** Int. J. Control 12, 721—738.
36. *Sakrison, D.J.* (1967). **The use of stochastic approximation to solve the system identification problem,** IEEE Trans. AC-12, 563—567.
37. *Saridis, G.N.* şi *G. Stein* (1968). **Stochastic approximation algorithms for linear discrete time system,** IEEE Trans. AC-13, 515—523.
38. *Saridis, G.N.* (1974). **Comparison of six on-line identification algorithm,** Automatica, 10, 69—79.
39. *Sen, A.* şi *N.K. Sinha* (1975). **A generalized pseudoinverse algorithm for unbiased parameter estimation,** Int. J. Syst. Sci 6, 1103—1109.
40. *Söderström, T.* (1973). **An on-line algorithm for approximate maximum likelihood identification of linear dynamic systems,** Report 7308, Drept. of Automatic Control, Lund Inst. of Tech. Lund.
41. *Söderström, T.* (1974 a). **Convergence of identification method based on the instrumental variable approach,** Automatica, 10, 685—688.
42. *Söderström, T.* (1974 b). **Convergence properties of the generalized least squares identification method,** Automatica 10, 617—626.
43. *Söderström T., L. Ljung* şi *I. Gustravsson* (1974). **A comparative study of recursive identification method,** Report 7427, Dept. of Automatic Control, Lund Inst. of Tech., Lund.
44. *Söderström, T., L. Ljung* şi *I. Gustavsson* (1978). **A theoretical analysis of recursive identification methods,** Automatica, 14, 231—244.
45. *Stewart, G.W.* (1973). **Introduction to matrix computations, Academic Press, New York.**
46. *Talmon, J.L.* şi *A.J.W. van den Boom* (1973). **On the estimation of transfer function parameters of processes and noise dynamics using a single stage estimator,** Proc. 3rd IFAC Symposium on Identification and System Parameter Estimation, Haga.
47. *Tertisco, M.* şi *P. Stoica* (1980). **Identificarea şi estimarea parametrilor sistemelor,** Editura Academiei, Bucureşti.
48. *Thornton, C.L.* şi *G.J. Bierman* (1978). **Filtering and error analysis via the UDU covariance factorization,** IEEE Trans. AC-23, 901—907.
49. *Young, P.C.* (1968). **The use of linear regression and related procedures for the identification of dynamic processes,** Proc. 7th IEEE Symposium on Adaptive Processes, UCLA.
50. *Young, P.C.* (1970). **An instrumental variable method for real time identification of a noisy process,** Automatica, 6, 271—287.
51. *Young, P.C.* (1974). **Recursive approaches to time series analysis,** Bull. Inst. Math. Applic. 10, 209—224.
52. *Young, P.C.* (1976). **Some observations on instrumental variable methods of time-series analysis,** Int. J. Control, 23, 593—612.
53. *Wieslander, I.* (1969). **Real time identification,** Report 6908, Dept. of Automatic Control, Lund Inst. of Tech., Lund.
54. *Wong, K.Y.* şi *E. Polak* (1967). **Identification of linear discrete time system using the instrumental variable method,** IEEE Trans. AC-12, 707—718.

BIMASC — PACHET DE SUBPROGRAME FORTRAN PENTRU MODELAREA, ANALIZA, PROIECTAREA ȘI SIMULAREA SISTEMELOR AUTOMATE

Dr. ing. *A. Varga*
I.T.C.I.

1. Introducere

În ultimii ani, pe plan mondial se observă o activitate susţinută pentru dezvoltarea unor algoritmi de calcul performanţi şi siguri, destinaţi rezolvării problemelor numerice specifice teoriei sistemelor automate. Se dispune în prezent de algoritmi eficienţi pentru principalele probleme de calcul ce intervin în modelarea, analiza, proiectarea şi simularea sistemelor automate multivariabile, cum sînt de exemplu analiza structurală [1], [2]; optimizarea liniar-pătratică [3]—[5]; alocarea polilor [6], [7]; simulare [8]—[12]. O sinteză comparativă amplă asupra algoritmilor de bază din domeniile de mai sus este făcută în ciclul de lucrări [13]—[15].

Datorită dezvoltărilor importante obţinute în domeniul algoritmilor, s-a început organizarea unor colecţii sistematice de programe ce implementează algoritmii cei mai performanţi. Acest lucru a fost posibil datorită disponibilităţii unor pachete de programe matematice de înaltă calitate, cum sînt BLAS — pentru problemele de bază ale algebrei liniare [16], LINPACK — pentru rezolvarea sistemelor de ecuaţii liniare [17] şi EISPACK — pentru probleme de valori şi vectori proprii [18], [19]. Aceste pachete au servit ca modele de organizare, programare şi documentare, pachetelor cu specific de automatică.

În perioada 1981—1982, s-a elaborat în cadrul Institutului Central pentru Conducere şi Informatică pachetul BIMAS destinat rezolvării problemelor numerice de bază ale analizei, proiectării şi simulării sistemelor automate liniare [20], [21]. BIMAS constituie o colecţie sistematică de subprograme FORTRAN portabile, elaborate pe baza standardelor de programare utilizate în LINPACK şi EISPACK. De altfel, o serie de rutine din aceste pachete au fost preluate fără modificări în BIMAS.

BIMASC este un pachet nou ce extinde pachetul BIMAS cu o serie de funcţii specifice analizei, modelării, proiectării şi simulării sistemelor multivariabile [22]. Algoritmii implementaţi în BIMASC au fost riguros selectaţi în vederea satisfacerii unor criterii de calitate cum sînt generalitate, siguranţă, stabilitate numerică, precizie şi eficienţă. Majoritatea subprogramelor din BIMASC apelează subprograme din pachetele BIMAS, EISPACK, LINPACK şi BLAS. BIMASC include, de asemenea, două pachete puternice pentru integrarea ecuaţiilor diferenţiale ordinare RKF45 [11] şi LSODE [12]. BIMASC

conţine 47 de subprograme originale şi 21 subprograme preluate din pachetele EISPACK, LINPACK, RKF45 şi LSODE.

BIMAS şi BIMASC au foarte multe caracteristici comune. Astfel majoritatea algoritmilor implementaţi utilizează transformări ortogonale în vederea asigurării unor proprietăţi numerice cît mai bune şi siguranţă cît mai ridicată. În cazul utilizării unor transformări neortogonale, de regulă, se determină informaţii asupra condiţionării numerice a problemelor. Ambele pachete folosesc o abordare structurală pentru rezolvarea problemelor complexe. Această abordare se reflectă printr-o înaltă modularitate a pachetelor. Majoritatea calculelor complexe se rezolvă printr-o secvenţă de apeluri la mai multe subprograme. Una dintre consecinţele abordării modulare este uşurinţa segmentării programelor. Astfel, pachetele se pot utiliza în condiţii bune chiar pe minicalculatoare. Toate subprogramele din BIMAS şi BIMASC sînt scrise în FORTRAN IV, în dublă precizie. Utilizarea unui FORTRAN standard, fără exploatarea unor facilităţi de lucru oferite de implementări particulare ale compilatoarelor, a condus la o portabilitate ridicată a pachetelor şi permite implementarea lor fără dificultăţi pe diferite tipuri de calculatoare.

Pachetele BIMAS şi BIMASC sînt disponibile pe minicalculatoarele româneşti I-100, 102F şi CORAL 4011, 4030, sub sistemele de operare MIX sau RSX-11M. Pachetele dispun de un set complet de programe de test. Aceste programe au fost astfel elaborate încît permit rezolvarea unei game largi de probleme concrete de analiză şi proiectare a sistemelor automate.

BIMASC acoperă cu subprograme principalele probleme de calcul ce intervin într-o metodologie structurală completă de analiză şi proiectare a sistemelor automate prin metode în spaţiul stărilor. Funcţiile pachetului BIMASC sînt prezentate în patru secţiuni distincte dedicate problemelor de modelare (secţ. 2), analiză (secţ. 3), proiectare (secţ. 4) şi simulare (secţ. 5).

2. Modelarea sistemelor liniare

Scopul modelării, în accepţiunea din această lucrare, este determinarea modelului adecvat utilizării metodelor de analiză, proiectare şi simulare. Majoritatea acestor metode utilizează descrieri de stare liniare, continue sau discrete, de forma

$$\lambda x(t) = Ax(t) + Bu(t) \quad (1)$$
$$y(t) = Cx(t)$$

unde x, u şi y sînt vectorul de stare n-dimensional, vectorul comenzilor m-dimensional şi respectiv vectorul ieşirilor p-dimensional, iar λ este operatorul diferenţial d/dt în cazul unui sistem continuu, sau operatorul de anticipare $\lambda x(t) = x(t+1)$ în cazul unui sistem discret. Sistemul (1) va fi referit de asemenea prin tripletul (A, B, C).

O altă descriere care poate servi ca bază de plecare pentru analiză şi proiectare este un model intrare-ieşire de forma

$$Y(\lambda) = G(\lambda)\, U(\lambda) \quad (2)$$

în care U (λ) şi Y (λ) sînt transformatele vectorilor de intrare şi respectiv ieşire, iar G (λ) este o matrice de transfer $p \times m$. În cazul continuu se utilizează, transformata Laplace, în locul lui λ utilizîndu-se de regulă variabila complexă s, iar în cazul discret, se foloseşte transformata Z, iar în loc de λ se întrebuinţează o variabilă complexă notată cu z. Matricea de transfer corespunzătoare modelului (1) este dată de formula:

$$G(\lambda)=C(\lambda I-A)^{-1}B \tag{3}$$

În cazul discret, elementele matricii de tansfer sînt funcţii raţionale. În cazul continuu, elementele matricii de transfer pot fi atît funcţii raţionale cît şi funcţii raţionale multiplicate cu factori de forma $e^{-\tau_{ij}s}$, în care τ_{ij} reprezintă timpul mort corespunzător canalului de la intrarea j la ieşirea i.

2.1. Transformări de similaritate

Vom spune că două sisteme (A, B, C) şi (A′, B′ C′) sînt *similare* dacă există o matrice de transformare nesingulară Q astfel încît

$$A'=Q^{-1}AQ,\ B'=Q^{-1}B,\ C'=CQ \tag{4}$$

Sistemele (A, B, C) şi (A′, B′, C′) legate prin (4) au aceeaşi matrice de transfer. Transformarea de forma (4) se numeşte *transformare de similaritate.*

Din punct de vedere numeric sîntem interesaţi să folosim numai transformări bine condiţionate, în sensul că *numărul de condiţionare* al matricii de transformare

$$\varkappa(Q)=\|Q\|\cdot\|Q^{-1}\| \tag{5}$$

nu este prea mare (este întotdeauna supraunitar). Pentru siguranţa calculelor, este de dorit ca pentru toţi algoritmii care utilizează transformări de similaritate, să dispunem de estimări ale numărului de condiţionare ale matricilor de transformare utilizate.

Transformări de similaritate ortogonale. O matrice nesingulară care satisface relaţia $Q^T Q=I$ se numeşte *ortogonală.* Transformările ortogonale au proprietăţi numerice remarcabile. În primul rînd ele sînt perfect condiţionate avînd $\varkappa(Q)=1$ şi din acest motiv utilizarea lor nu modifică de regulă sensibilitatea problemelor. În al doilea rînd o serie de algoritmi bazaţi pe utilizarea transformărilor ortogonale sînt *numeric stabili*, în sensul că matricile *calculate* A′ B′ şi C′ prin similaritate sînt exacte pentru un sistem care este foarte apropiat de sistemul (A, B, C). Matematic acest lucru se poate scrie prin egalităţile

$$A'=Q^T(A+\delta A)Q,\quad B'=Q^T(B+\delta B),\quad C'=(C+\delta C)Q \tag{6}$$

în care δA, δB şi δC satisfac condiţii de forma

$$\|\delta A\|\leqslant\varepsilon\|A\|,\quad \|\delta B\|\leqslant\varepsilon\|B\|,\quad \|\delta C\|\leqslant\varepsilon\|C\| \tag{7}$$

În inegalităţile de mai sus, ε este un factor generic egal cu un multiplu modest al preciziei relative a maşinii.

Cîteva rutine din BIMAS se pot folosi pentru reducerea lui A la forme particulare: *forma Hessenberg*, *forma Schur reală* (FSR), sau FSR ordonată.

Transformările ortogonale rezultate cu aceste rutine, sau transformările ortogonale arbitrare se pot aplica unui sistem (A, B, C) utilizînd subrutina ORTEQ.

Multe dintre problemele de calcul ce intervin în modelarea și analiza sistemelor liniare necesită reducerea sistemului (1) prin transformări de similaritate ortogonale la *forma standard de controlabilitate* (FSC)

$$A'=\begin{bmatrix} A_R & X \\ 0 & A_N \end{bmatrix} \quad B'=\begin{bmatrix} B_R \\ 0 \end{bmatrix} \quad C'=[C_R \ C_N] \tag{8}$$

în care A_R, B_R și C_R au dimensiunile $k\times k$, $k\times m$ și $p\times k$, respectiv. În general, dacă nu există nici o transformare de similaritate de forma (4) care reduce sistemul (A, B, C) la forma (8) cu $0\leqslant k<n$, atunci vom spune că sistemul (1) este *controlabil*. Deoarece controlabilitatea este o proprietate structurală a perechii (A, B), vom spune echivalent că perechea (A, B) este controlabilă. În cazul cînd sistemul (A, B, C) se poate pune sub forma (8) cu $k<n$, vom spune că sistemul (1) este *necontrolabil* sau echivalent, perechea (A, B) este necontrolabilă. Un sistem în FSC (8) are proprietatea că perechea (A_R, B_R) este controlabilă, iar sistemul (A_R, B_R, C_R) are aceeași matrice de transfer ca sistemul (A, B, C). A_N este partea necontrolabilă a matricii de stare.

Reducerea unui sistem la FSC se poate face utilizînd subrutina TSCO, care opțional acumulează transformările ortogonale. Algoritmul implementat în TSCO este bazat pe metoda descrisă în [2], în care calculele de rang bazate pe descompunerea valorilor singulare s-au înlocuit cu calcule de rang bazate pe descompunerea QR cu pivotare.

Forma standard de observabilitate (FSO) a unui sistem (A, B, C) se poate obține de asemenea utilizînd transformări de similaritate ortogonale. Matricile sistemului în FSO au forma

$$A'=\begin{bmatrix} A_R & 0 \\ X & A_N \end{bmatrix} \quad B'=\begin{bmatrix} B_R \\ B_N \end{bmatrix} \quad C'=[C_R \ 0] \tag{9}$$

în care A_R, B_R, și C_R au dimensiunile $k\times k$, $k\times m$ și $p\times k$, respectiv. Dacă $k=n$ vom spune că sistemul (1) este *observabil* sau echivalent perechea (C, A) este observabilă. Dacă $k<n$, sistemul este *neobservabil*. În FSO, perechea (C_R, A_R) este observabilă, iar sistemul (A_R, B_R, C_R) are aceeași matrice de transfer ca sistemul inițial. A_N este în acest caz partea neobservabilă a matricii de stare.

Reducerea sistemului (1) la FSO se poate face utilizînd TSCO pentru *sistemul dual* (A^T, C^T B^T). Sistemul dual se poate construi cu subrutina DUALS.

Transformări de similaritate neortogonale. Cîteva rutine din BIMAS efectuează transformări neortogonale pentru reducerea lui A la forme particulare (Hessenberg, bloc-diagonală). Aplicarea transformărilor de similaritate arbitrare, neortogonale, se poate face în cazul general cu subrutina SIMEQ. Această rutină determină de asemenea și o estimare a numărului de condiționare a matricii de transformare.

Deseori modelele de stare minimale (controlabile și observabile) obținute din reprezentări de tip intrare-ieșire sînt slab scalate datorită unei alegeri improprii a sistemului de coordonate pentru variabilele de stare. Astfel de scalări conduc de regulă la balansări nedorite în model în raport cu gradul de

controlabilitate (GC) și gradul de observabilitate (GO) ale sistemului. Din punct de vedere calitativ, GC și GO reprezintă măsuri ale depărtării unui sistem dat de unul necontrolabil, sau neobservabil, respectiv. Balansarea unui sistem urmărește maximizarea simultană și egalizarea GC și GO ale unui sistem utilizînd transformări de similaritate.

O tehnică de balansare de acest fel a fost propusă de Moore [25]. În acest caz GC și GO ale sistemului se definesc ca fiind cele mai mici valori singulare ale Grammian-ului de controlabilitate și respectiv observabilitate. Procedura de balansare asigură egalitatea și forma diagonală a matricilor Grammian ale sistemului balansat. Un sistem cu această proprietate se numește sistem *balansat intern* (SBI).

O procedură pentru calculul transformărilor de balansare pentru sisteme stabile a fost propusă de Laub [26] și este implementată în subrutina TSBALI. Modelul SBI calculat cu TSBALI se poate utiliza pentru construirea unor modele aproximative de ordin redus pentru sistem. O altă metodă de balansare, propusă de Varga [27], este implementată în subrutina TSBAL. Sistemul balansat rezultat are matricea de stare în FSR, iar matricile Grammian sînt ortogonal similare cu aceeași matrice diagonală pozitiv definită. Ambele rutine estimează numerele de condiționare ale matricilor de transformare utilizate

2.2. Construcția realizărilor de stare ale matricilor de transfer

Metodele de identificare experimentale ale proceselor determină de cele mai multe ori modele de tip intrare-ieșire, continue sau discrete, specificate prin matrici de transfer. Dacă matricea de transfer G (λ) a unui sistem este cunoscută și are numai elemente funcții raționale, atunci putem construi mai multe realizări de stare (A, B, C) care satisfac (3) utilizînd una dintre subrutinele RENEM, RENEMC sau RENEMO. Modelul de stare construit cu fiecare dintre aceste rutine are forma

$$A=\begin{bmatrix} A_1 & 0 & \dots & 0 \\ 0 & A_2 & \dots & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ 0 & 0 & \dots & A_k \end{bmatrix} \qquad B=\begin{bmatrix} B_1 \\ B_2 \\ \cdot \\ \cdot \\ \cdot \\ B_k \end{bmatrix} \tag{10}$$

$$C=[C_1 \ C_2 \ \dots \ C_k]$$

în care subsistemele $(A_i\ B_i,\ C_i)$, $i=1, 2, \dots, k$ sînt controlabile și observabile. RENEM construiește în general modele de stare necontrolabile și neobservabile, RENEMC construiește modele controlabile, iar RENEMO construiește modele observabile. În cazul cînd sistemul este stabil, se poate face o balansare utilă la nivelul fiecărui subsistem utilizînd subrutina BALRNM. Realizări de stare particulare pentru sistemele cu o intrare și o ieșire se pot construi cu subrutinele REMC1 și REMO1.

Determinarea unui model de stare *minimal* (controlabil și observabil) pentru o matrice de transfer rațională se face de regulă în două etape. În prima

etapă se construieşte o realizare neminimală de forma (10) utilizînd RENEM, RENEMC sau RENEMO. În etapa a doua se calculează modelul de stare minimal utilizînd subrutina REMIN. REMIN apelează TSCO pentru eliminarea succesivă a părţilor necontrolabile şi neobservabile ale sistemului neminimal. Această procedură de calcul a realizărilor minimale este numeric stabilă. Trebuie să menţionăm că în foarte multe cazuri concrete, modelul de stare construit cu RENEMC sau RENEMO rezultă minimal, etapa a doua nefiind necesară.

2.3. Evaluarea matricilor de transfer din reprezentări de stare

Pentru o reprezentare de stare (A, B, C) dată a unui sistem, sîntem cîteodată interesaţi să calculăm matricea de transfer corespunzătoare G (λ) (3), sau a unei linii, a unei coloane sau chiar a unui singur element al acestei matrici. În acest scop se poate utiliza una dintre subrutinele TSMT sau TSMT2. TSMT implementează un algoritm propus în [28]. Acest algoritm utilizează transformări de similaritate ortogonale pentru a determina realizări de stare minimale, pe rînd pentru fiecare canal intrare-ieşire al sistemului multivariabil. Apoi, se evaluează pentru fiecare element polii, zerourile şi factorul de amplificare. Calculul polilor şi zerourilor are la bază determinări de valori proprii.

TSMT2 utilizează în locul transformărilor ortogonale din TSMT transformări de similaritate stabilizate. De asemenea, calculul polilor şi zerourilor este înlocuit cu evaluări de polinoame caracteristice ale unor matrici Hessenberg. În general, TSMT2 este mult mai rapid decît TSMT şi de regulă şi mai precis. Cu toate acestea, precizia *garantată* apriori pentru TSMT este mai mare decît pentru TSMT2 şi acest lucru se poate manifesta efectiv în cazul unor exemple particulare.

2.4. Discretizarea sistemelor continue

Proiectarea legilor de reglare pentru conducerea numerică directă cu calculator a proceselor continue necesită utilizarea unor modele discrete (sau eşantionate) pentru proces. De asemenea, simularea sistemelor liniare continue se poate efectua eficient prin utilizarea modelului de stare discretizat.

Considerăm sistemul liniar continuu

$$\frac{dx(t)}{dt} = Ax(t) + Bu(t) \tag{11}$$

şi fie h perioada de eşantionare. Presupunem că u are valoarea constantă u (k) pe fiecare interval [kh, (k+1) h), k=0, 1, ... şi notăm x (k) valoarea lui x zentru t=kh. În acest caz, sistemul discretizat corespunzător sistemului (11) este dat de

$$x(k+1) = E\,x(k) + Fu(k) \tag{12}$$

unde

$$E = e^{Ah}, \quad F = \int_0^h e^{At}\,Bdt \tag{13}$$

iar e^X este exponenţiala matricii X definită prin dezvoltarea în serie

$$e^X = 1 + \frac{X}{1!} + \frac{X^2}{2!} + \dots$$

Dacă definim matricea

$$D = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix}$$

atunci E şi F din (13) se pot determina din identitatea

$$e^{Dh} = \begin{bmatrix} E & F \\ 0 & I \end{bmatrix} \tag{14}$$

Această metodă de discretizare a fost propusă de Van Loan [8] şi este implementată în subrutina TSCD. Exponenţiala (14) se evaluează utilizînd o metodă ce combină o procedură de bloc diagonalizare propusă în [24] şi o procedură dezvoltată în [29] pentru evaluarea exponenţialei matriciale prin aproximări raţionale Padé. Această metodă propusă în [27] este prezentată şi în lucrarea mai accesibilă [14].

Discretizarea matricilor de transfer a sistemelor continue se poate face utilizînd subrutina TMTCD. În cazul cînd matricea de transfer conţine elemente cu timpi morţi, printr-o alegere convenabilă a perioadei de eşantionare putem elimina factorii iraţionali din model. Algoritmul implementat în TMTCD se aplică în parte fiecărui element al matricii de transfer, deci îl vom schiţa doar în cazul unei singure funcţii de transfer. Pentru partea raţională a funcţiei de transfer se construieşte o reprezentare de stare minimală, care se discretizează utilizînd aproximări raţionale Padé pentru exponenţiala matricială [29]. Funcţia de transfer a sistemului discretizat se calculează cu TSMT2. Dacă funcţia de transfer continuă are timp mort, atunci numitorul funcţiei de transfer discrete se multiplică cu factorul corespunzător întîrzierii datorate timpului mort.

Trebuie să facem observaţia că ambele metode de discretizare implementate în TSCD şi TMTCD utilizează un extrapolator de ordin zero pe post de convertor discret-continuu al semnalelor de intrare ale sistemului continuu.

3. Analiza sistemelor liniare

Analiza unui sistem are ca scop studierea unor proprietăţi ale sistemului (stabilitate, controlabilitate/stabilizabilitate, observabilitate/detectabilitate), determinarea unor mărimi caracteristice (poli, zerouri, margine de stabilitate), studierea comportării dinamice a sistemului în buclă deschisă. În acest capitol ne vom referi la primele două dintre aspectele de mai sus. Studierea comportării dinamice a sistemului prin simulare va fi discutată în secţ. 5.

3.1. Calculul polilor şi zerourilor sistemului

Cunoaşterea polilor este importantă în studierea dinamicii sistemului şi a stabilităţii sale. Pentru sistemul (1) *polii* sînt valorile proprii ale matricii de stare A. Sistemul (1) este stabil dacă toate valorile proprii ale lui A au părţi reale negative în cazul continuu sau au modulele mai mici decît unu în cazul discret. *Marginea de stabilitate* pentru sistemul (1) se poate defini ca partea reală maximă (în cazul continuu) sau ca modulul maxim (în cazul discret) a valorilor proprii ale matricii A. Subrutina TSTA determină polii şi marginea de stabilitate ale sistemului şi testează condiţia de stabilitate.

Zerourile unui sistem joacă un rol important în teoria reglării sistemelor liniare [30], [31]. Pentru definirea zerourilor unui sistem considerăm descrierea de stare mai generală dată de ecuaţiile

$$\lambda x(t) = Ax(t) + Bu(t) \tag{15}$$

$$y(t) = Cx(t) + Du(t)$$

în care, spre deosebire de modelul (1), apare o legătură directă intrare-ieşire în cazul cînd matricea D este nenulă.

Pentru sistemul descris de ecuaţiile (15) *zerourile sistemului* sau *zerourile invariante* se definesc ca zerourile Smith ale fasciculului de matrici

$$S(\lambda) = \begin{bmatrix} \lambda I - A & B \\ -C & D \end{bmatrix} \tag{16}$$

Matricea S(λ) de dimensiune $(n+p)\times(n+m)$ se numeşte *matricea sistem* [32]. Zerourile Smith ale lui S(λ) sînt zerourile (incluzînd multiplicităţile) celui mai mare divizor comun al tuturor minorilor identic nenuli de ordin $n+\min(m, p)$ ai lui S(λ). Dacă $p=0$, zerourile sistemului se numesc *zerouri de decuplare la intrare* (z.d.i.) şi sînt egale cu polii necontrolabili ai sistemului, iar cînd $m=0$, zerourile se numesc *zerouri de decuplare la ieşire* (z.d.o.) şi sînt egali cu polii neobservabili ai sistemului. Cînd sistemul este minimal (controlabil şi observabil), zerourile sistemului se numesc *zerouri de transmisie* şi sînt egale cu zerourile McMillan [32] ale matricii de transfer a sistemului

$$G(\lambda) = C(\lambda I - A)^{-1}B + D \tag{17}$$

O metodă eficientă de calcul a zerourilor sistemului a fost propusă de Emami-Naeini şi Van Dooren [33] şi se bazează pe forma canonică Kronecker a matricii sistem S(λ). Acest algoritm extrage din fasciculul de matrici (16) un fascicul regular $\lambda F - E$ ($\det(\lambda F - E) \not\equiv 0$) ale cărui valori proprii generalizate sînt zerourile finite ale lui S(λ). Extragerea fasciculului regular se face utilizînd în exclusivitate transformări ortogonale şi este implementară în subrutina MZEROS (adaptată din [33]). Valorile proprii generalizate se calculează cu rutine din BIMAS.

3.2. Analiza controlabilităţii şi observabilităţii

Pentru testarea controlabilităţii sistemului (1), se poate utiliza subrutina TSCO. Sistemul este controlabil dacă partea sa controlabilă în FSC (8) are ordinul n. O altă posibilitate este utilizarea subrutinei MZEROS pentru determinarea z.d.i. ale fasciculului

$$S(\lambda)=[\lambda I-A \quad B]$$

Dacă $S(\lambda)$ nu are nici un z.d.i. atunci sistemul (1) este controlabil.

Pentru testarea observabilităţii sistemului, aceleaşi subrutine se utilizează pentru perechea duală (A^T, C^T). Cînd se utilizează MZEROS, sistemul este observabil dacă fasciculul

$$S(\lambda)=\begin{bmatrix}\lambda I-A\\ -C\end{bmatrix}$$

nu are z.d.o.

Sistemul (1) este *stabilizabil* dacă polii necontrolabili (valorile proprii ale lui A_N din (8)) sînt stabili şi este *detectabil* dacă polii neobservabili (valorile proprii ale lui A_N din (9)) sînt stabili. Pentru testarea stabilizabilităţii sau detectabilităţii se pot utiliza rezultatele calculate fie de TSCO sau MZEROS. Dacă se utilizează TSCO, testul se va face cu subrutine TSTA aplicat matricii A_N din (8) sau (9). Dacă se utilizează MZEROS atunci se va testa stabilitatea z.d.i. sau z.d.o. calculate.

4. Proiectarea sistemelor de reglare

Considerăm sistemul liniar descris de ecuaţiile

$$\begin{aligned}\lambda x(t)&=Ax(t)+Bu(t)+Ew(t)\\ y(t)&=Cx(t)\\ y_r(t)&=C_r x(t)\\ e(t)&=r(t)-y_r(t)\end{aligned} \tag{18}$$

în care x, u şi y sînt ca mai înainte, w este vectorul l-dimensional al perturbaţiilor măsurabile, iar y_r, r şi e sînt vectorii q-dimensionali ai ieşirilor reglate, referinţelor şi erorilor de reglare, respectiv. Vom presupune că ieşirile reglate sînt măsurabile, deci matricea C_r are forma $C_r=[I \;\; 0]\,C$, unde I este o matrice unitate de ordin q.

Compensatorul este un sistem liniar care are ca intrări ieşirea măsurată y, referinţele r şi eventual perturbaţiile măsurabile w şi generează ca ieşire comanda u care se aplică procesului. Compensatorul poate fi descris de exemplu de ecuaţiile:

$$\begin{aligned}\lambda z(t)&=Fz(t)+Gy(t)+Kw(t)+Lr(t)\\ u(t)&=H_1 z(t)+H_2 y(t)\end{aligned} \tag{19}$$

în care z este vectorul de stare al compensatorului. Există diverse structuri de compensator care să satisfacă aceleaşi cerinţe de proiectare. Astfel, unele dintre matricile K, L sau H_2 din (19) pot fi identic nule. În alte situaţii, în expresia lui u în (19) pot apare termeni suplimentari, aşa cum este în cazul acţiunii directe de la perturbaţii sau de la referinţe.

Problema de reglare se poate formula în felul următor: să se determine matricile compensatorului (19) astfel încît: 1) sistemul în buclă închisă să fie stabil; 2) eroarea de reglare staţionară să fie nulă, adică $e(t) \to 0$ pentru $t \to \infty$ (anularea erorii staţionare) şi 3) proprietăţile de stabilitate şi de anulare a erorii staţionare să se menţină în prezenţa unor perturbaţii mici, dar arbitrare în parametrii sistemului sau ai compensatorului. Proprietatea 3) se mai numeşte *robusteţe*, iar compensatorul (19) ce satisface 1), 2) şi 3) se numeşte *compensator robust*. Problema de reglare are soluţie dacă [30], [31]: 1) sistemul (A, B, C) este stabilizabil şi detectabil, şi 2) zerourile sistemului (A, B, C_r) nu coincid cu frecvenţele care formează spectrul semnalelor exterioare (referinţe şi perturbaţii). De exemplu, în cazul cel mai frecvent în practică, al unor referinţe constante şi a unor perturbaţii de tip treaptă, condiţia asupra zerourilor este ca sistemul că nu aibă zerouri nule în cazul continuu, sau zerouri egale cu unitatea, în cazul discret. Pentru acest exemplu, compensatorul robust este o generalizare a regulatorului tradiţional PI folosit curent în practica industrială.

Proiectarea unui compensator robust implică stabilizarea unui sistem extins format din sistemul (18) cuplat cu un *model intern* [30] sau *servocompensator* [31]. Modelul intern este descris de ecuaţia

$$\lambda x_i(t) = A_i x_i(t) + B_i e(t) \tag{20}$$

în care x_i este starea modelului intern. Matricea A_i se alege astfel încît să includă o reduplicare de q ori a dinamicii semnalelor externe. Perechea (A_i, B_i) se alege controlabilă. De exemplu modelul intern discret pentru semnale exterioare de tip treaptă are forma simplă

$$x_i(t+1) = x_i(t) + e(t)$$

Sistemul extins care trebuie stabilizat este dat de ecuaţia

$$\lambda x'(t) = A'x(t) + B'u(t) \tag{21}$$

în care vectorul de stare x' se poate alege

$$x' = \begin{bmatrix} x_i \\ x \end{bmatrix} \text{ sau } x' = \begin{bmatrix} x \\ x_i \end{bmatrix} \tag{22}$$

iar perechile (A', B') corespunzătoare sînt date de (23) sau (24), respectiv:

$$A' = \begin{bmatrix} A_i & -B_i C_r \\ 0 & A \end{bmatrix} \quad B' = \begin{bmatrix} 0 \\ B \end{bmatrix} \tag{23}$$

$$A' = \begin{bmatrix} A & 0 \\ -B_i C_r & A_i \end{bmatrix} \quad B' = \begin{bmatrix} B \\ 0 \end{bmatrix} \tag{24}$$

Procedura de proiectare generală a compensatorului robust are următoarele etape [31], [34]:

1) Se formează sistemul extins (21) cu matricile A' și B' date de (23) sau (24);

2) Se stabilizează sistemul (21) cu o reacţie de stare de forma

$$u(t)=H'x'(t)=H_1x_1(t)+Hx(t) \tag{25}$$

unde $H'=[H_1\ H]$ sau $H'=[H\ H_1]$, după cum s-a utilizat extinderea (23) sau (24), respectiv;

3) Se proiectează un estimator de stare pentru sistemul care are ca intrări y, u şi eventual w şi furnizează x_e, starea estimată a sistemului.

În BIMASC există o serie de subprograme cu ajutorul cărora proiectarea compensatorului se poate efectua. Pentru construirea matricilor A' şi B' din (23) sau (24) se utilizează subrutina EXTI. Această subrutină poate genera automat matricile A_1 și B_1 pentru clasa semnalelor externe de tip treaptă sau rampă. Subprogramele pentru calculul matricii de reacţie după stare sînt prezentate în § 4.1, iar proiectarea estimatoarelor este considerată în § 4.2. În § 4.3. se prezintă subrutine pentru proiectarea unor regulatoare cu acţiune directă (feed-forward).

4.1. Proiectarea regulatoarelor cu reacţie de la stare

În proiectarea compensatoarelor prin metoda prezentată mai înainte apare frecvent următoarea problemă de calcul: să se determine matricea de reacţie de la starea H din legea de comandă liniară

$$u(t)=Hx(t) \tag{26}$$

astfel încît matricea de stare a sistemului în buclă închisă, A+BH, să aibă anumite proprietăţi dorite. În continuare vom prezenta trei metode alternative care se pot utiliza pentru calculul matricii de reacţie.

Stabilizarea sistemelor liniare. Fie S o mulţime simetrică de numere complexe definită ca

$$S=\{\lambda : \mathrm{Re}(\lambda) < a\} \tag{27}$$

pentru sisteme continue sau

$$S=\{\lambda : |\lambda| < a\} \tag{28}$$

pentru sisteme discrete. *Problema de stabilizare* cere să se determine H astfel încît toate valorile proprii ale lui A+BH să aparţină lui S. Condiţia necesară şi suficientă pentru existenţa soluţiei acestei probleme este ca sistemul (18) să fie stabilizabil în raport cu S, adică să nu aibe z.d.i. în S. Matematic acest lucru se poate scrie sub forma:

$$\mathrm{rang}\,[\lambda I - A \quad B]=n \tag{29}$$

pentru toţi $\lambda \in S$.

Doi algoritmi de stabilizare sînt implementaţi în subrutinele STAC şi STAD pentru stabilizarea sistemelor continue respectiv, discrete. Ambii algoritmi separă spectrul lui A în raport cu partiţionarea planului complex determinată

de S. Separarea spectrului are la bază ordonarea FSR matricii A și delimitează partea instabilă a sistemului în raport cu S. Stabilizarea sistemului se face numai pentru partea sa instabilă de regulă de ordin mult mai mic decît ordinul sistemului. Condiția de stabilizabilitate (29) se testează implicit și se determină și o estimare a gradului de controlabilitate a părții instabile a sistemului. Descrierea amănunțită a algoritmilor implementați se găsește în lucrările [35], [15].

Pentru proiectarea legii de reglare (25) cu subrutinele STAC sau STAD, se recomandă construirea extinderii (24) cu EXTI.

Alocarea polilor. Fie R o mulțime simetrică de n numere complexe în raport cu axa reală. *Problema de alocare a polilor* cere să se determine H din (26) astfel încît spectrul lui $A+BH$ să fie R. Dacă dorim ca anumite valori proprii „bune" ale lui A să rămînă nemodificate, atunci evident le vom include în R. Condiția necesară și suficientă pentru existența soluției problemei de alocare a polilor este ca toate valorile proprii care se modifică să aparțină părții controlabile a sistemului. Matematic acest lucru necesită satisfacerea condiției (29) de toți λ aparținînd acelei părți a spectrului lui A care se modifică.

Alocarea polilor se poate efectua utilizînd subrutina SALOC, care implementează un algoritm propus de Varga [2], [15]. Acest algoritm este implementat într-o manieră care permite păstrarea nealterată a polilor care aparțin mulțimii S definite prin (27) sau (28). Acest lucru este realizat separînd spectrul lui A, prin intermediul formei sale Schur, în raport cu partiționarea planului complex determinată de mulțimea S. Polii se alocă printr-o procedură iterativă. La fiecare interație se alocă unu sau doi poli. Polii complecși se alocă în perechi complex conjugate. Matricea de reacție H rezultă ca o sumă de matrici componente, fiecare alocînd unu sau doi poli. Pentru $m>1$, norma acestor matrici componente este minimizată. Dacă toate aceste norme sînt de ordinul lui $\|A\|/\|B\|$, atunci algoritmul efectuează calculele într-o manieră numeric stabilă [36]. Condițiile de controlabilitate pentru polii modificați se testează implicit în algoritm, iar polii necontrolabili, dar stabili, sînt automat incluși printre polii care nu se modifică. Pentru toți polii modificați, SALOC evaluează gradele de controlabilitate ale acestora. Pe lîngă matricea H ce alocă polii sistemului, SALOC determină și FSR a matricii de stare $A+BH$, ca și matricea de transformare ortogonală corespunzătoare. Această facilitate se poate exploata în proiectarea estimatoarelor de stare.

Pentru proiectarea legii de reglare (25) cu subrutina SALOC, se recomandă, pentru creșterea preciziei calculelor, construirea extinderii (23) cu EXTI.

Optimizarea liniar-pătratică. Matricea de reacție de la stare H se poate determina optimală astfel încît legea de comandă liniară (26) să minimizeze un indice de performanță pătratic de forma

$$\int_0^{\infty} [x(t)^T Qx(t)+u(t)^T R\, u(t)]\, dt \tag{30}$$

pentru un sistem continuu, sau

$$\sum_{t=0}^{\infty} [x(t)^T Qx(t)+u(t)^T R\, u(t)] \tag{31}$$

pentru un sistem discret. În mod uzual se presupune că matricile de ponderare Q și R din (30) și (31) satisfac condițiile $Q \geqslant 0$ și $R > 0$.

Rezolvarea problemei de optimizare liniar-pătratică presupune rezolvarea unei ecuaţii matriciale algebrice Riccati (EMAR) de forma

$$A^TX+XA-XBR^{-1}B^TX+Q=0 \tag{32}$$

în cazul continuu, sau de forma

$$X=A^TXA-A^TXB(R+B^TXB)^{-1}B^TXA+QB \tag{33}$$

în cazul discret. O condiţie necesară şi suficientă pentru existenţa şi unicitatea unei soluţii pozitiv definite a ecuaţiilor (32) sau (33) este ca perechea (A, B) să fie stabilizabilă, iar perechea $(A, \sqrt{Q})$ să fie detectabilă, unde $\sqrt{Q}$ este rădăcina pătrată matricială al lui Q.

Dacă cunoaştem soluţia X a EMAR, atunci matricea de reacţie optimală H rezultă cu formulele

$$H=R^{-1}B^TX \tag{34}$$

în cazul continuu, sau

$$H=-(R+B^TXB)^{-1}B^TXA \tag{35}$$

în cazul discret. Pentru evaluarea acestor formule se poate folosi subrutina OPTR.

Rezolvarea EMAR se poate face utilizînd mai multe subrutine din BIMAS. Metoda interativă Newton este implementată în subrutinele NTNC şi NTND (incluse de asemenea în BIMASC) pentru rezolvarea EMAR continue şi respectiv discrete. Ambele rutine evaluează şi matricea de reacţie optimală. Metoda Newton necesită iniţializarea procesului interativ cu o matrice de reacţie stabilizatoare. În acest scop, se pot utiliza fie SALOC, fie STAC, sau STAD. Algoritmul implementat este descris în [37].

Metode directe de rezolvare a EMAR sînt implementate în subrutinele EXTC şi SCHV pentru sisteme continue şi EXTD şi SCHV pentru sisteme discrete cu matricea de stare inversabilă. SCHV implementează metoda propusă de Laub [3]. Pentru sisteme discrete cu matricea de stare singulară, se pot utiliza subrutinele EXT2 şi GSCHV. GSCHV a fost elaborată pe baza metodei propuse de Van Dooren [5]. O prezentare a acestor metode se găseşte în lucrarea [15].

Notă : subrutina OPTR şi subrutinele pentru rezolvarea EMAR au fost programate de dr. ing. V. Sima.

4.2. Proiectarea estimatoarelor de stare

Implementarea legilor de comandă după stare de forma (25) sau (26) necesită cunoaşterea vectorului de stare sau a unei estimaţii x_e a acestui vector. Un estimator de stare de ordin „n" pentru sistemul (18) este descris de ecuaţiile

$$\lambda z(t)=Fz(t)+Gy(t)+Bu(t)+Ew(t) \tag{36}$$

$$x_e(t)=z(t)$$

Din condiţia ca eroarea de estimare

$$e(t)=x(t)-x_e(t) \quad (37)$$

să se anuleze pentru $t \to \infty$, rezultă următoarea expresie pentru matricea de stare a estimatorului:

$$F=A-GC \quad (38)$$

în care G trebuie astfel determinat încît F să aibă valori proprii stabile. Dacă perechea (A, C) este detectabilă, matricea G se poate calcula prin oricare dintre metodele descrise în secţiunea 3.1, utilizînd în locul perechii (A, B), perechea duală (A^T, C^T). Dacă se utilizează SALOC, matricea rezultată F^T este în FSR. SALOC determină de asemenea matricea de transformare ortogonală. Dacă s-a determinat în prealabil matricea de reacţie H din (25) sau (26), atunci se poate obţine, printr-o similaritate ortogonală aplicată sistemului (F, [G, B, E], H) utilizînd ORTEQ, ca matricea F să rezulte în FSR. Această proiectare a compensatorului format din ecuaţiile (25) şi (36) conduce la o reducere semnificativă a numărului de operaţii cerut de implementarea „on-line" a estimatorului. O reducere suplimentară a numărului de operaţii rezultă prin bloc-diagonalizarea lui F.

Dacă se utilizează tehnica de optimizare liniar-pătratică pentru determinarea lui G, atunci matricile de ponderare care apar în EMAR asociată, au semnificaţii precise. Q este matricea de covarianţă a zgomotului de la intrare w (t), iar R este matricea de covariantă a zgomotului de măsurare. Estimatorul rezultat se numeşte *filtru Kalman-Bucy* în cazul continuu, sau *predictor Kalman* în cazul discret.

Deşi simplu de proiectat şi de implementat, estimatorul de ordin „n" are o anumită redundanţă, deoarece determină o estimare a întregului vector de stare, cînd de fapt o parte a acestui vector se poate calcula direct din ieşirile măsurate. Astfel, pentru sistemul (18), putem utiliza un estimator de stare minimal de ordin $n-p$ ($p=\text{rang } C$) descris de ecuaţiile

$$\begin{aligned} \lambda z(t) &= Fz(t)+Gy(t)+Lu(t)+Kw(t) \\ x_e(t) &= My(t)+Nz(t) \end{aligned} \quad (39)$$

O procedură eficientă de proiectare a unui estimator de stare prin alocarea polilor a fost propusă de Varga [34]. Această procedură este implementată în subrutina SAESTM. Estimatorul rezultat are matricea de stare în FSR. O reducere a lui F la forma bloc-diagonală se poate efectua printr-o transformare de similaritate aplicată sistemului (F, [G, L, K], N).

4.3. Proiectarea regulatoarelor cu acţiune directă

În cazul unui sistem stabil asupra căruia acţionează semnale exterioare (referinţe şi perturbaţii) constante, regulatorul cu acţiune directă de la referinţe şi perturbaţii este descris de relaţia

$$u(t)=H_r r(t)+H_d w(t) \quad (40)$$

H_r şi H_d se determină cu formulele

$$H_r=-[G_r(s)]^{\#} \qquad H_d=[G_r(s)]^{\#}\, G_d(s) \quad (41)$$

în care # desemnează pseudo-inversa, $G_r(\lambda)$ și $G_d(\lambda)$ sînt matrici de transfer date de

$$G_r(\lambda)=C_r(\lambda I-A)^{-1}B, \quad G_d(\lambda)=C_r(\lambda I-A)^{-1}E \tag{42}$$

În (41), $s=0$ pentru un sistem continuu și $s=1$ pentru un sistem discret.

În cazul cînd modelul sistemului este descris exact de ecuațiile (18), regulatorul cu acțiune directă (40) asigură eroare staționară de reglare nulă. Totuși de regulă acest regulator nu se poate folosi singur, deoarece asupra proceselor acționează și perturbații nemăsurabile ale căror efecte nu se pot elimina decît prin utilizarea unui regulator cu reacție.

Pentru proiectarea regulatoarelor cu acțiune directă se pot utiliza subrutinele MTFF și STFF. MTFF calculează H_r și H_d direct din matricele de transfer (42), iar STFF calculează aceste matrici pe baza modelului de stare (18).

5. Simularea sistemelor

Simularea sistemelor are un dublu scop. În primul rînd, simularea este un puternic instrument de analiză a comportării dinamice a sistemelor. Prin simulare se pot obține o serie de informații cantitative și calitative utile despre proces, cum sînt : caracterul regimului tranzitoriu (oscilant sau amortizat), valorile suprareglajelor la modificarea unor referințe, durata regimului tranzitoriu la acțiunea unor perturbații, evidențierea interconexiunilor existente în sistem. De asemenea, prin simulare se poate verifica acuratețea unor aproximații făcute la deducerea modelelor cum sînt : liniarizarea unor modele neliniare, reducerea ordinului sau discretizarea unor modele continue. În al doilea rînd, simularea este un auxiliar prețios în proiectarea sistemelor de comandă, deoarece este modalitatea prin care putem evalua performanțele obținute în urma proiectării sau putem compara performanțele diferitelor structuri de comandă în vederea fixării structurii celei mai simple de implementat, care satisface specificațiile de proiectare.

Subprogramele prezentate în acest capitol nu efectuează simulări, ci constituie mai degrabă instrumente utile ce se pot utiliza în programe de simulare a unor sisteme neliniare sau liniare. Aceste programe se vor elabora de utilizatori în funcție de scopul concret urmărit.

5.1. Integrarea numerică a ecuațiilor diferențiale

Considerăm problema integrării ecuațiilor diferențiale ordinare (EDO) de ordin întîi cu condiții inițiale

$$\frac{dx(t)}{dt}=F(t,\ x),\quad x(t_0)=x_0,\qquad t_0\leqslant t\leqslant t_f \tag{43}$$

în care x este un vector cu n componente, iar F (t, x) este o funcție vectorială pe asemenea cu n componente. Pentru integrarea numerică a sistemului (43)

am luat în considerare două clase de metode. Pentru sisteme cu dinamică uniformă (Jacobianul $J=\partial F/\partial x$ are valori proprii cu ordine de mărime apropiate) se utilizează metode Runge-Kutta-Fehlberg de ordinul 4—5. Această metodă este implementată în cadrul pachetului RKF45, una dintre implementările cele mai performante şi sigure existente în momentul de faţă [11]. Pentru integrarea EDO „*stiff*“, pentru care Jacobianul lui F are o dispersie foarte mare a valorilor proprii, se utilizează metode implicite în mai mulţi paşi. Unul dintre pachetele cele mai puternice existente la ora actuală pe plan mondial este pachetul LSODE, care implementează două metode de integrare cu pas şi ordin variabile : metode Adams de ordin pînă la 12 pentru EDO ne-„stiff“ şi metoda diferenţelor inverse (metoda Gear) de ordin pînă la 5, pentru probleme „stiff“. Trebuie să subliniem că ambele pachete au mai multe opţiuni de control a erorilor de integrare şi permit luarea în considerare cu uşurinţă a unor structuri particulare a unor probleme. De exemplu LSODE poate trata distinct probleme care au matricea Jacobian în formă de bandă. Aceste rutine stau la baza metodelor de simulare implementate în pachetul CASAD [23].

5.2. Simularea structurilor de comandă liniare

Considerăm sistemul liniar constant descris de ecuaţiile (18). Simularea sistemului (18) în buclă deschisă se face deseori presupunînd anumite forme de semnal particulare (treaptă, rampă) pentru u, w sau r. În cazul simulării sistemului în buclă închisă, comanda se calculează sub forma :

$$u(t)=u_1(t)+u_2(t) \tag{44}$$

unde u_1 este termenul de reacţie, iar u_2 este termenul de acţiune directă. Ambii termeni în (44) sînt opţionali.

Termenul de reacţie u_1 se poate genera în una din următoarele forme :

$$\begin{aligned} u_1(t)&=Hx_e(t)\\ u_1(t)&=Hy_r(t)\\ u_1(t)&=H_ix_i(t)\\ u_1(t)&=Hx_e(t)+H_ix_i(t)\\ u_1(t)&=Hy_r(t)+H_ix_i(t) \end{aligned} \tag{45}$$

În (45) x_e este fie starea sistemului (dacă nu se foloseşte estimator de stare) fie estimaţia vectorului de stare generată de un estimator de ordin „n“ (36) sau un estimator minimal (39) ; x_i este vectorul de stare al modelului intern (20), în care eroarea se poate calcula în unul din următoarele moduri

$$\begin{aligned} e(t)&=r(t)-y_r(t)\\ e(t)&=y_m(t)-y_r(t) \end{aligned} \tag{46}$$

În (46), y_m este ieşirea unui model de referinţă descris de

$$\begin{aligned} \lambda x_m(t)&=A_mx_m(t)+B_mr(t)\\ y_m(t)&=C_mx_m(t) \end{aligned} \tag{47}$$

Termenul cu acţiune directă din (44) poate avea una dintre expresiile următoare:

$$u_2(t)=H_r r(t)$$

$$u_2(t)=H_d w(t) \tag{48}$$

$$u_2(t)=H_r r(t)+H_d w(t)$$

Ecuaţiile (18), (36) sau (39), (20), (44) — (48) formează împreună un sistem extins de forma

$$\lambda v(t)=A'v(t)+E'w(t)+E''r(t) \tag{49}$$

$$y_r(t)=C'v(t)$$

în care v este vectorul de stare format din vectorii de stare x, z, x_m şi x_i.

Pentru simularea sistemului (49) se pot utiliza două abordări. Prima abordare se bazează pe formarea matricilor A', E' şi E'' din (49) şi discretizarea sistemului (49) pentru o perioadă de eşantionare aleasă de utilizator. Această metodă este considerată şi analizată în lucrarea [14]. Deşi are cîteva caracteristici atractive (simularea discretă este foarte uşoară), această abordare are unele inconveniente, cum sînt: dificultăţi în calculul exponenţialelor matriciale pentru sisteme mari sau perioade de eşantionare mari, lipsa controlului asupra erorilor, nu permite exploatarea structurii particulare a unor probleme.

A doua abordare constă în utilizarea şi adaptarea programelor de uz general pentru integrarea numerică a EDO [9], cum sînt RKF45 sau LSODE. Această abordare permite un control riguros al erorilor şi exploatarea structurii particulare a unor probleme. Utilizarea acestei abordări presupune scrierea de către utilizator a rutinei de evaluare a derivatelor (membrul drept din prima ecuaţie (49)). În acest scop s-a elaborat rutina GSTEP, care evaluează (49) fără a forma explicit matricile A', E', E'' sau C'.

Simularea structurilor de comandă discrete se efectuează prin apelarea repetată a lui GSTEP. Dacă se utilizează RKF45 sau LSODE pentru simularea sistemelor continue, GSTEP stă la baza rutinei de evaluare a derivatelor. Pentru reducerea numărului de operaţii necesare evaluării de către GSTEP a derivatelor, matricile A, F, A_m şi A_i se pot reduce prin similaritate la forma Hessenberg înainte de primul apel al lui GSTEP. Pentru simularea sistemelor „stiff“ cu LSODE, se evaluează şi Jacobianul problemei — A', sau al unei aproximaţii a acestei matrici. O aproximare bloc diagonală a Jacobianului este descrisă în [10]. Aceste idei stau la baza metodelor de simulare implementate în pachetul CASAD [23]

5.3. Facilităţi grafice

Pentru reprezentarea grafică a mai multor funcţii de timp se poate utiliza subrutina AOGR, care permite folosirea unui set de caractere de reprezentare specificat de utilizator. Scările de reprezentare pot fi furnizate de utilizator sau calculate automat. Lăţimea reprezentării poate fi de 80 sau 132 coloane.

6. Concluzii

În această lucrare s-a prezentat un pachet, BIMASC, de subprograme FORTRAN destinat modelării, analizei, proiectării și simulării sistemelor automate multivariabile. De asemenea, s-au făcut unele referiri și la pachetul BIMAS, (care va fi prezentat într-o lucrare viitoare) destinat rezolvării unor probleme numerice de bază din domeniile de mai sus. BIMAS și BIMASC constituie o bază puternică de programe performante pentru proiectarea asistată de calculator a sistemelor automate. Pe baza acestor pachete s-a elaborat un puternic pachet interactiv de proiectare a sistemelor automate, denumit CASAD [23]. BIMAS, BIMASC și CASAD constituie a doua generație de programe cu caracter aplicativ (după pachetul SIPAC [38]) destinate problemelor de automatică, elaborate la noi în țară. Aceste pachete sînt la nivelul celor mai bune pachete existente pe plan mondial.

BIBLIOGRAFIE

1. *P. Van Dooren*, **The generalized eigenstructure problem in linear system theory**, IEEE Trans. Autom. Control, vol. AC-26, p. 111—129, 1981.
2. *A. Varga*, **Numerically stable algorithm for standard controllability form determination**, Electronics Letters, vol. 17, p. 74—75, 1981.
3. *A. J. Laub*, **A Schur method for solving the algebric matrix Riccati equations**, IEEE Trans. Autom. Control, vol. AC-24, p. 913—921, 1979.
4. *T. Pappas, A. J. Laub, N. R. Sandell*, **Oh the numerical solution of the discrete-time algebraic Riccati equation**, IEEE Trans. Autom. Control, vol. AC-25, p. 631—641, 1980.
5. *P. Van Dooren*, **A generalized eigenvalue approach for solving the Riccati equations**, Rep. NA-80.02, Comp. Scie. Dept., Stanford Univ., 1980.
6. *G. S. Miminis, C. C Paige*, **An algorithm for pole assignment of time-invariant multi-imput linear systems**, 21-st IEEE Conf. on Decision and Control, San Diego, 1982.
7. *A. Varga*, **A Schur method for pole assignment**, IEEE Trans. Autom. Control, vol. AC-26, p. 517—519, 1981.
8. *C. F. Van Loan*, **Computing integrals involving matrix exponențials**, IEEE Trans. Autom. Control, vol. AC-23, p. 395—404, 1978.
9. *W. Enright*, **On efficient and reliable numerical solution of large linear systems of ODE's**, IEEE Trans. Autom. Control, vol. AC-24, p. 905—908, 1979.
10. *A. Varga, V. Sima, C. Y. Varga*, **On numerical simulation of linear continuous control systems**, Preprints of SIMULATION'83 Symposium, Praga, iunie 1983.
11. *G. E. Forsythe, M. A. Malcolm, C. B. Moler*, **Computer methods for mathematical computations**, Prentice-Hall, Englewood Cliffs, 1977.
12. *A. C. Hindmarsh*, **LSODE and LSODI, two new initial value ordinary differential equation solvers**, ACM-Siqnum Newsletter, vol. 15, p. 10—11, 1980.
13. *A. Varga, V. Sima*, **Tehnici numerice de calcul in teoria sistemelor liniare : (I) Tehnici numerici de bază ale algebrei liniare**, AMC, vol. 34, p. 135—168, Editura tehnică, 1981.
14. *A. Varga*, **Tehnici numerice de calcul la teoria sistemelor liniare : (II) Analiza sistemelor liniare**, AMC, vol. 35, p. 29—61, Editura tehnică, 1983.
15. *A. Varga, V. Sima*, **Tehnici numerice de calcul in teoria sistemelor liniare : (III) Proiectarea sistemelor automate**, AMC, vol. 36, p. 5—49, Editura tehnică, 1983.
16. *C. Lawson, R. Hanson, D. Kincaid, F. Krogh*, **Basic linear algebra subprograms for Fortran usige**, ACM Trans. Math. Software, vol. 5, p. 303—323, 1979.

17. *J.J. Dongarra, J.R. Bunch, C.B. Moler, G.W. Stewart*, **LINPACK User's guide**, SIAM, Philadelphia, 1979.
18. *B.T. Smith, J.M. Boyle, J.J. Dongarra, B.S. Garbow, Y. Ikebe, V.C. Klema, C.B. Moler*, **Matrix eigensystem routines – EISPACK guide**, Lect. Notes Comp. Scie, vol. 6, Springer Verlag, Berlin, 1974.
19. *B.S. Garbow, J.M. Boyle, J.J. Dongarra, C.B. Moler*, **Matrix eigensystem routines – EISPACK guide extension**, Lect. Notes Comp. Scie, vol. 51, Springer Verlag, Berlin, 1977.
20. *A. Varga, V. Sima*, **BIMAS – a basic mathematical package for computer-aided systems analysis and design**, Preprints of VII-th IFAC World Congress, Budapesta, iunie, 1984 (Report ICI, TR-01.83, 1983).
21. *A. Varga, V. Sima*, **BIMAS – Genereal Description**, Report ICI, TR-03.82, 1982
22. *A. Varga*, **BIMASC – General Description**, Report ICI, TR-01.83, 1983.
23. *A. Varga*, **CASAD – General Description**, Report ICI, TR-12.83, 1983.
24. *C.A. Bavely, G.W. Stewart*, **An algorithm for computing reducing subspaces by block diagonalization**, SIAM J. Numer. Anal., vol. 10, p. 359–367, 1979.
25. *B.C. Moore*, **Principal component analysis in linear systems : controllability, observability and model reduction**, IEEE Trans. Autom. Control, vol. AC-26, p. 17–32, 1981.
26. *A.J. Laub*, **On computing „balancing" transformations**, Preprints of JACC Symp., San Francisco, August, 1980.
27. *A. Varga*, **Contribuţii la conducerea robustă cu calculator a proceselor continue**, Teză de doctorat, I.P. Bucureşti, 1981.
28. *A. Varga, V. Sima*, **A numerically stable algorithm for transfer function matrix evaluation**, Int. J. Control, vol. 33, p. 1 123–1 133, 1981.
29. *R.C. Ward*, **Numerical computation of the matrix experimential with accuracy estimate**, SIAM J. Numer. Anal., vol. 14, p. 600–610, 1977.
30. *W.M. Wonham*, **Linear multivariable control. A Geometric approach**, Springer Verlag, Berlin, 1979.
31. *E.J. Davison, A. Goldenberg*, **The robust control of a general servomechanism problem : the servocompensator**, Automatica, vol. 11, p. 461–471, 1975.
32. *H.H Rosenbrock*, **State-space and multivariable theory**, Nelson, 1970.
33. *A. Emami-Naeini, P. Van Dooren*, **Computation of zeros of linear multivariable systems**, Automatica, vol. 18, p. 415–430, 1982.
34. *A. Varga*, **Computer aided design of robust compensators by pole assignment**, Preprints of SOCOCO'82 Symp. Madrid, sept. 1982.
35. *A. Varga*, **On stabilization algorithms for linear time-invariant systems**, Rev. Roum. Scie. Techn. – Electrotechn. et Energ., vol. 26, p. 115–124, 1981.
36. *A. Varga*, **The numerical stability of an algorithm for pole assignment**, Preprints of IFAC Symp. on CAD of Multivariablie Tehnological Systems, Purdue University, West Lafayette, sept., 1982.
37. *V. Sima*, **On the real Schur form in linear control system design**, Rev. Roum. Scie. Techn. – Electrotech. et Energ., vol. 25, p. 625–632, 1980.
38. *Th. Popescu, V. Sima, A. Varga, C. Vasiliu*, SIPAC – **Sistem de programe pentru identificarea proceselor şi proiectarea asistată de calculator a sistemelor automate. Manual de utilizare.** ICI, Lab. Calc. Proces, 1978.

PROIECTAREA-OPTIMIZAREA ASISTATĂ DE CALCULATOR A PROCESELOR TEHNOLOGICE ÎN CONSTRUCŢIA DE MAŞINI. SISTEMUL PAPT-1

Nicolae-Valentin Ivan
Universitatea din Braşov

1. Introducere

Competitivitatea unui anumit produs, pe orice fel de piaţă, este tot mai mult hotărîtă de costurile legate de tehnologie. Prin urmare, acesteia trebuie să i se acorde o atenţie aparte, în cadrul eforturilor generale care urmăresc sporirea în tot mai mare măsură a efectelor economice favorabile, concomitent cu asigurarea unei anumite tehnicităţi, astfel că, punîndu-se problema asigurării calităţii dorite a produselor la un cost cît mai scăzut, proiectarea-optimizarea proceselor tehnologice se impune cu necesitate. Optimizarea proceselor tehnologice, avîndu-se în vedere aspectele legate de complexitate şi metodă, constituie o problemă dificilă. Soluţia trebuie desprinsă din formularea completă a problemei şi cunoscîndu-se punctul de vedere ales pentru optimizare. Desigur, formularea problemei vizează aspectul matematic al acesteia şi astfel apare necesitatea modelării matematice. Formularea problemei în sens matematic necesită efectuarea unei analize atente a sistemelor tehnologice de prelucrare [12, 13, 2].

Abordarea sistemică a problemei optimizării proceselor tehnologice este necesară pentru evidenţierea multitudinii de factori şi aspecte în interdependenţele lor, precum şi pentru corelarea corespunzătoare a datelor în vederea rezolvării problemei.

Sub formă de schemă-bloc, sistemul tehnologic de prelucrare este prezentat în fig. 1 [12, 13]. Trebuie precizat faptul că fig. 1 se referă la prelucrările pe maşinile-unelte aşchietoare, domeniu vizat de prezenta lucrare.

Ceea ce trebuie remarcat în mod deosebit în fig. 1 este existenţa fluxului informaţional, care, în cazul optimizării, capătă o importanţă primordială, ştiut fiind faptul că informaţia este ingredientul esenţial al conducerii [4] (optimizarea trebuie privită drept componentă a unui proces de conducere). Şi astfel fiind vorba de optimizarea proceselor tehnologice, are sens să se vorbească despre sistemul informaţional asociat [12, 13].

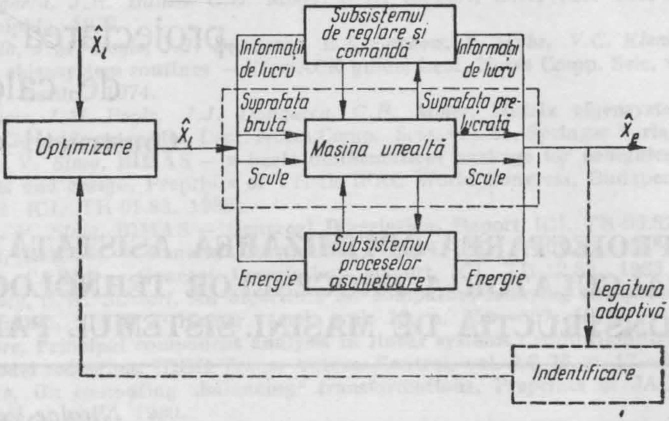


Fig. 1. Sistemul tehnologic de prelucrare.

Generalizînd [42], prin sistemul informațional asociat optimizării proceselor tehnologice trebuie să se înțeleagă ansamblul informațiilor, surselor și nivelelor consumatoare, canalelor de circulație, procedurilor și mijloacelor de tratare a informațiilor, astfel structurat încît să permită, prin intermediul unor reguli, aplicarea unor metode în vederea obținerii deciziei tehnologice optime (fig. 2 [2]).

Figurile 1, 2 evidențiază cele două moduri fundamentale de optimizare a proceselor tehnologice [35, 3, 12, 13]: optimizarea *externă* și optimizarea *nternă*, diferențiate prin tehnica de rezolvare precum și prin modul în care *i*sînt considerate informațiile.

Prezenta lucrare se înscrie pe linia optimizării externe a proceselor tehnologice de prelucrare pe mașinile-unelte așchietoare, caz în care, cu referire la fig. 2, informațiile de ieșire se referă la documentația tehnologică. Ansamblul informațiilor de intrare vizează precizia piesei, caracteristicile și mulțimea procedeelor așchietoare, echipamentele tehnologice, date contabile etc.

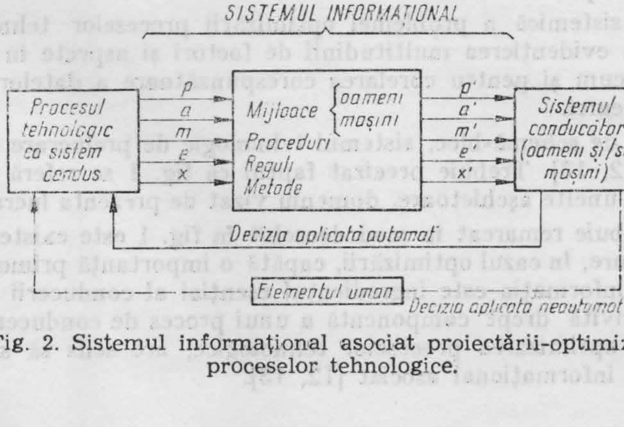


Fig. 2. Sistemul informațional asociat proiectării-optimizării proceselor tehnologice.

2. Necesitatea utilizării calculatoarelor electronice în proiectarea-optimizarea proceselor tehnologice de prelucrare prin aşchiere

Motivul principal pentru care procesele tehnologice se pretează a fi proiectate, optimizate şi în general conduse cu calculatoare este faptul că informaţia, în asemenea activităţi, deţine rolul esenţial [44].

Implicarea calculatoarelor electronice în proiectarea-optimizarea proceselor tehnologice rezolvă două mari probleme. Prima se referă la prelucrarea, în timp util, a unui volum foarte mare de informaţii, iar a doua la rezolvarea unui număr relativ mare de subprobleme conexe, fiecare avînd mai multe grade de libertate. Avîndu-se în vedere optimizarea în sensul actual al noţiunii, ea nici n-ar putea fi făcută fără aportul calculatoarelor electronice care, numai ele, permit implementarea teoriilor moderne de optimizare. Tocmai de aceea optimizarea (deci şi proiectarea-optimizarea) reprezintă una din funcţiile de bază ale calculatoarelor electronice.

3. Formularea problemei

Pentru formalizarea matematică a problemei proiectării-optimizării unui proces tehnologic este necesară asocierea unor variabile caracteristice precum şi a relaţiilor dintre ele, în scopul reflectării legităţilor proprii, a proprietăţilor care privesc destinaţia procesului precum şi a criteriului de optimizare. Modelul matematic astfel obţinut [13] va permite vehicularea după o anumită logică şi o anumită tehnică a imaginilor informaţionale (fig. 2) despre elementele sistemului tehnologic (fig. 1).

Generic, modelul matematic al problemei de proiectare-optimizare se poate exprima printr-un sistem de relaţii ce consemnează interdependenţele existente între cele n variabile asociate (dependente sau nu de timp), adică:

$$\begin{cases} f(x_1, x_2, \ldots, x_j, \ldots, x_n)=0, & j=1, 2, \ldots, n \\ r_i(x_1, x_2, \ldots, x_j, \ldots, x_n) \gtreqless 0, & i=1, 2, \ldots, m \end{cases} \tag{1}$$

în care: f este expresia analitică a criteriului de optimizare, numită funcţie-obiectiv, funcţie-scop, funcţie-performanţă sau funcţie-eficienţă; r_i — cele m restricţii sau constrîngeri ale problemei, reflectînd legităţile fizice precum şi anumite conexiuni artificiale între variabile, anume create şi impuse de destinaţia procesului (pe scurt, condiţiile concrete de funcţionare).

În mod corect, modelul matematic (1) reprezintă de fapt un sistem de modelare. Este preferabilă o asemenea modelare matematică, deoarece exprimarea problemei printr-un singur model matematic practic n-ar fi posibilă, datorită complexităţii aspectelor ce trebuie avute în vedere la proiectare-optimizare, precum şi datorită interdependenţelor strînse dintre ele.

Cea mai importantă problemă care trebuie rezolvată, este desigur determinarea variantei tehnologice optime. În conformitate cu ideea considerării acesteia ca o problemă de programare pseudobooleeană [1], particularizarea modelului matematic (1) duce la [1, 2, 8, 9]:

$$\begin{cases} \min CX = \min \left(\sum_{j=1}^{n} c_j x_j \right), \; j \in N = \{1, 2, \ldots, n\} \\ F_i(X) \text{ rel } 0, \; i \in L = \{1, 2, \ldots, l\} \\ x_j = 1 \text{ sau } 0 \end{cases} \tag{2}$$

unde : CX este funcţia obiectiv a problemei ; $F_i(X)$ — cele l condiţii concrete (constrîngerile problemei) ; rel — oricare din operatorii relaţionali : $<$, $\leqslant$, $=$, $>$, $\geqslant$; X — vectorul coloană care descrie mulţimea celor n procedee avută iniţial în vedere ; C — vectorul linie ale cărui componente precizează costurile prelucrării piesei prin procedeele corespunzătoare.

Vectorul X este dat de :

$$X = [x_1 \, x_2 \, x_3 \ldots x_n]' \tag{3}$$

ale cărui componente sînt variabile bivalente de cod, asociate procedeelor de prelucrare ($x_j = 1$ sau 0 după cum procedeul codificat prin variabila x_j este sau nu optim).

Componentele vectorului C :

$$C = [c_1 \, c_2 \, c_3 \ldots c_n] \tag{4}$$

exprimă costurile prelucrării piesei prin procedeele corespunzătoare [12, 13, 27, 8, 23].

Prin dezvoltare, partea restrictivă a modelului matematic (2) aferentă prelucrării unei porţiuni K din piesă devine [12, 13, 5, 6] :

$$\begin{cases} \Sigma x_j = 1, \; j \in N_K \\ \Sigma T_{ej} \cdot x_j \leqslant T_{P_K} \\ \Sigma R_{ej} \cdot x_j \leqslant R_{P_K} \end{cases} \tag{5}$$

unde : N_K este submulţimea indicilor variabilelor care codifică procedeele avute iniţial în vedere, aferente prelucrării porţiunii K din piesă (aceste procedee se află într-o relaţie de autoexcludere) ; T_{ej}, R_{ej} — toleranţele, respectiv rugozităţile economice asociate procedeului codificat prin x_j ; T_{P_K}, R_{P_K} — toleranţa respectiv rugozitatea prescrise porţiunii K a piesei.

Fiind vorba de o problemă de programare matematică în numere întregi, din categoria celor de decizii binare, după obţinerea soluţiei [33, 34, 35] în procesul tehnologic vor trebui utilizate acele procedee pentru care $x_j = 1$.

Exprimarea analitică a funcţiei-obiectiv din modelul matematic (2) presupune cunoaşterea regimurilor de aşchiere şi a adaosurilor de prelucrare, acestea fiind alte două importante subprobleme care trebuie rezolvate în cadrul

proiectării-optimizării proceselor tehnologice de prelucrare pe mașinile-unelte așchietoare.

După cum este cunoscut, determinarea regimurilor de așchiere face parte din categoria problemelor de programare matematică neliniară [14]. Prin urmare, modelul matematic (1) particularizat în acest caz este:

$$\min C(Y) \mid a_i(Y) \leqslant 0,\ Y > 0,\ i=\{1, 2, \ldots, r\} \quad (6)$$

unde vectorul Y este:

$$Y=[v\ s]' \text{ sau } Y=[v_1 s_1\ v_2 s_2 \ldots v_q s_p]' \quad (7)$$

după cum este vorba de prelucrări cu scule singulare sau cu multiscule, și unde componentele vectorului Y reprezintă parametrii regimurilor de așchiere pentru cele q scule (viteza v de așchiere și avansul s).

C (Y) respectiv a_i (Y) reprezintă funcția-obiectiv (costul), respectiv cele r restricții [14, 27, 8, 11, 7, 13, 12, 30].

În conformitate cu așa-numita „metodă a planului de secțiune" a lui Kelley [4], mulțimea programelor problemei (6) poate fi reprezentată ca intersecția unui număr suficient de mare de semispații care o conțin. Aceste semispații pot fi generate cunoscîndu-se un vector inițial [14, 12]:

$$Y^{\circ}=[v^{\circ}\ s^{\circ}]' \text{ sau } Y^{\circ}=[v_1^{\circ} s_1^{\circ} \ldots v^{\circ} s^{\circ}]' \quad (8)$$

și asociindu-se inegalitățile:

$$\varphi_i(Y, Y^{\circ})=a_i(Y^{\circ})+[Y-Y^{\circ}]' \nabla a_i(Y^{\circ}) \leqslant 0 \quad (9)$$

pentru fiecare restricție neliniară și:

$$\psi(Y, Y^{\circ})=C(Y^{\circ})+[Y-Y^{\circ}]' \nabla C(Y^{\circ})-Z \leqslant 0 \quad (10)$$

pentru funcția obiectiv, dacă este neliniară, în care: $\nabla a_i(Y^{\circ})$, $\nabla C(Y^{\circ})$ sînt gradienții funcțiilor respective pentru vectorul inițial Y°; Z este o nouă variabilă introdusă reclamată de metoda de rezolvare [4] și care trebuie acuma minimizată, devenind de fapt funcție-obiectiv.

Modelul matematic care trebuie rezolvat devine:

$$\min Z \mid \varphi_i(Y, Y^{\circ}) \leqslant 0,\ \psi(Y, Y^{\circ}) \leqslant 0,\ Y>0,\ i=\{1, \ldots, r\} \quad (11)$$

Problema (11) se rezolvă în mai multe etape. Pentru fiecare etapă modelul fiind liniar se poate aplica algoritmul SIMPLEX [4, 15], algoritm care este cel mai corespunzător pentru domeniul programării matematice liniare [38].

În ceea ce privește adaosul de prelucrare, aspectul este și el destul de complicat, chiar dacă se are în vedere utilizarea calculatorului electronic. Mai ales din punctul de vedere al datelor inițiale, deci al băncii de date, apare complicația. Metodologia de calcul tradițională [41, 37] poate fi utilizată cu condiția algoritmizării adecvate și organizării corespunzătoare a datelor inițiale [18]. Pentru o exploatare avantajoasă a calculatorului se impune, în primul rînd, modelarea matematică a datelor inițiale experimentale [19], cunoscîndu-se faptul că acestea se prezintă în volum mare tabelar [41]. Adică, pentru parametri cum sînt: curbura specifică a semifabricatelor, erorile de fixare etc. trebuie stabilite formule empirice, în general de tipul:

$$y=ax^b;\ y=ax+b \quad (12)$$

unde y reprezintă parametrul supus modelării, iar x — parametrul independent, în general diametrul [19, 13]. Determinarea valorilor lui a și b se poate face prin metoda celor mai mici pătrate sau alte metode [43].

După cum rezultă din cele de mai sus, subproblemele conexe proiectării-optimizării proceselor tehnologice sînt deosebit de complexe în esență, complicația matematică amplificîndu-se atunci cînd este vorba de introducerea calculatoarelor electronice (evident, în stadiul de analiză). Este firesc să fie așa, deoarece utilizarea calculatoarelor electronice nu trebuie să se rezume la aceleași metode tradiționale de prelucrare a informațiilor. Pe de altă parte, trebuie subliniat în mod deosebit că succesul unui sistem informatic depinde în foarte mare măsură de banca de date, precum și de modul în care este concepută exploatarea sistemului. Legat de acest ultim aspect, trebuie amintit și faptul că minutul-calculator costă încă destul de mult. În sfîrșit, mai trebuie adăugat că succesul unui asemenea sistem este asigurat atunci cînd utilizatorul nu este necesar să se specializeze și în domeniul calculatoarelor.

De cele de mai sus s-a ținut cont la realizarea sistemului PAPT-1, care va fi tratat în cele ce urmează. În dorința de a optimiza exploatarea calculatorului, în ceea ce privește sistemul PAPT-1, în faza de analiză s-a realizat un studiu de preoptimizare [21, 31, 32], referitor la varianta tehnologică optimă precum și la regimurile și adaosurile optime de prelucrare. Doar rezultatele acestui studiu au fost introduse în programele operative ale sistemului PAPT-1. În acest fel sistemul PAPT-1 a devenit extrem de rentabil. O anumită experiență cîștigată în domeniu face oportună sublinierea necesității studiului de preoptimizare ori de cîte ori se pornește la proiectarea unui sistem informatic de concepție-optimizare a proceselor tehnologice.

4. Sistemul PAPT-1

Creat în cadrul catedrei de T.C.M. a Universității din Brașov, sistemul PAPT-1 (Proiectarea Automată a Proceselor Tehnologice de prelucrare pe strunguri automate multiax) [12, 21, 31, 26, 17, 29, 13] a fost conceput ca un subsistem grefat pe sectorul de pregătire a fabricației și permite obținerea documentației tehnologice, sub forma planului de operații, la imprimanta calculatorului Felix C-256, fără intervenția omului în fazele de luare a deciziilor tehnologice.

Ca sistem informatic, PAPT-1 este prezentat în fig. 3, de unde rezultă cu claritate lanțurile și procedurile de tratare a informațiilor.

Sistemul este astfel conceput, încît procesul tehnologic rezultă rapid funcție doar de un minimum de informații de intrare, strict legate de particularitățile geometrice și de precizie ale piesei ale cărui proces tehnologic interesează.

Experiența cîștigată prin proiectarea și exploatarea sistemului PAPT-1 arată că atunci cînd este vorba de utilizarea calculatoarelor electronice în rezolvarea problemelor tehnologice, este avantajos să se utilizeze tipizarea proceselor tehnologice [13, 28]. În acest sens trebuie precizat că sistemul PAPT-1 are în vedere familia de piese din fig. 4 și 5. Fig. 5 se referă la sub-

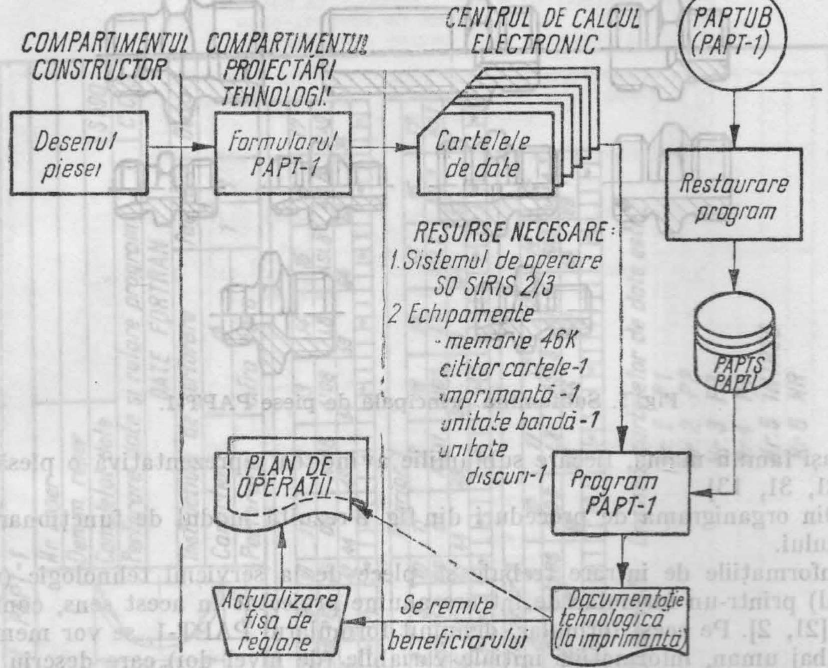


Fig. 3. Organigrama de proceduri a sistemului PAPT-1.

familia de bază cu care a început devenirea operativă a sistemului PAPT-1. Cu această ocazie trebuie menționat și faptul că aceeași experiență a demonstrat că pentru elaborarea și exploatarea unor asemenea sisteme informatice este preferabil să se lucreze cu mai multe subfamilii de piese, toate aparținînd

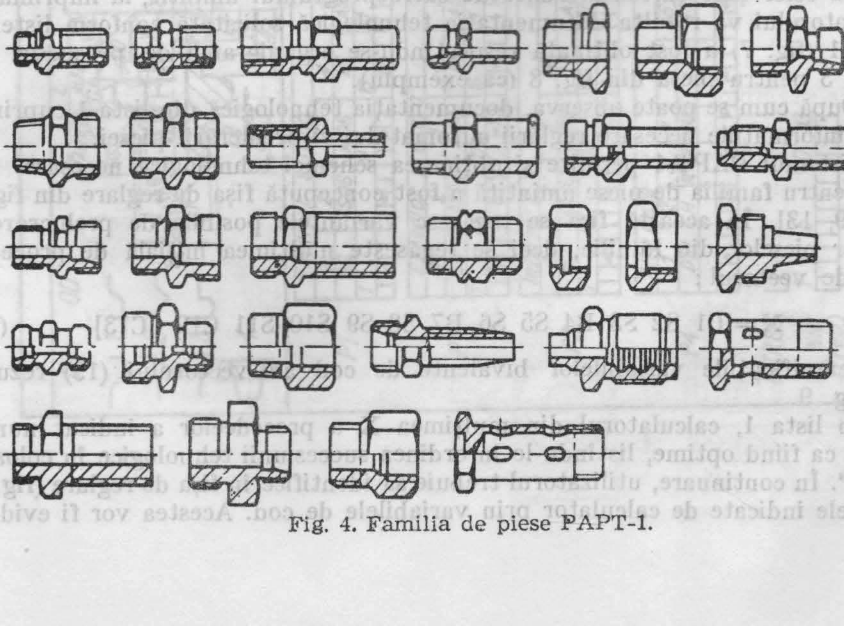


Fig. 4. Familia de piese PAPT-1.

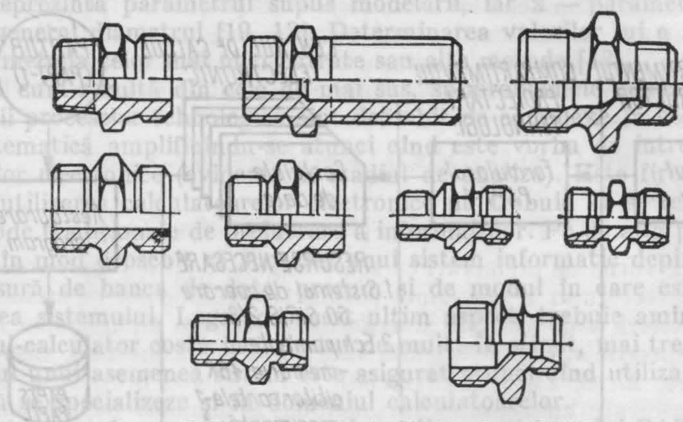


Fig. 5. Subfamilia principală de piese PAPT-1.

aceleiaşi familii mamă, fiecare subfamilie avînd ca reprezentativă o piesă-cod [12, 21, 31, 13].

Din organigrama de proceduri din fig. 3 rezultă modul de funcţionare a sistemului.

Informaţiile de intrare trebuie să plece de la serviciul tehnologic (utilizatorul) printr-un formular de intrare anume proiectat în acest sens, conform fig. 6 [21, 2]. Pe acest formular, denumit formularul PAPT-1, se vor memora, în limbaj uman, informaţiile iniţiale variabile (de nivel doi) care descriu geometria piesei precum şi condiţiile de precizie impuse acesteia. În continuare, aceste informaţii vor fi codificate pe cartele, obţinîndu-se astfel prin perforare şase cartele de date. Urmează apoi prelucrarea acestor date de către programul PAPT-1 (de fapt un sistem de subprograme conduse de un program principal), memorat pe bandă magnetică sau chiar pe discul-sistem. În urma prelucrării celor şase cartele de date de către programul amintit, la imprimanta calculatorului va rezulta documentaţia tehnologică solicitată conform listei 1. Lista 1 (fig. 7) a fost obţinută respectîndu-se organigrama de proceduri din figura 3 pentru piesa din fig. 8 (ca exemplu).

După cum se poate observa, documentaţia tehnologică din lista 1 cuprinde toate informaţiile necesare reglării automatului şi prelucrării piesei.

Sistemul PAPT-1 permite şi obţinerea schemei tehnologice necesare.

Pentru familia de piese amintită a fost concepută fişa de reglare din fig. 9 [21, 29, 13]. În această fişă se regăsesc variantele posibile de prelucrare a tuturor pieselor din familie, deci se regăseşte mulţimea iniţială de procedee dată de vectorul :

$$X=[C1\ S2\ S3\ B4\ S5\ S6\ B7\ S8\ S9\ S10\ S11\ C12\ \ C13]' \qquad (13)$$

Semnificaţiile variabilelor bivalente de cod ale vectorului (13) rezultă din fig. 9.

În lista 1, calculatorul din mulţimea X a procedeelor a indicat numai cîteva ca fiind optime, listîndu-le în ordinea succesiunii tehnologice în coloana „COD". În continuare, utilizatorul trebuie să identifice în fişa de reglare (fig. 9) schemele indicate de calculator prin variabilele de cod. Acestea vor fi eviden-

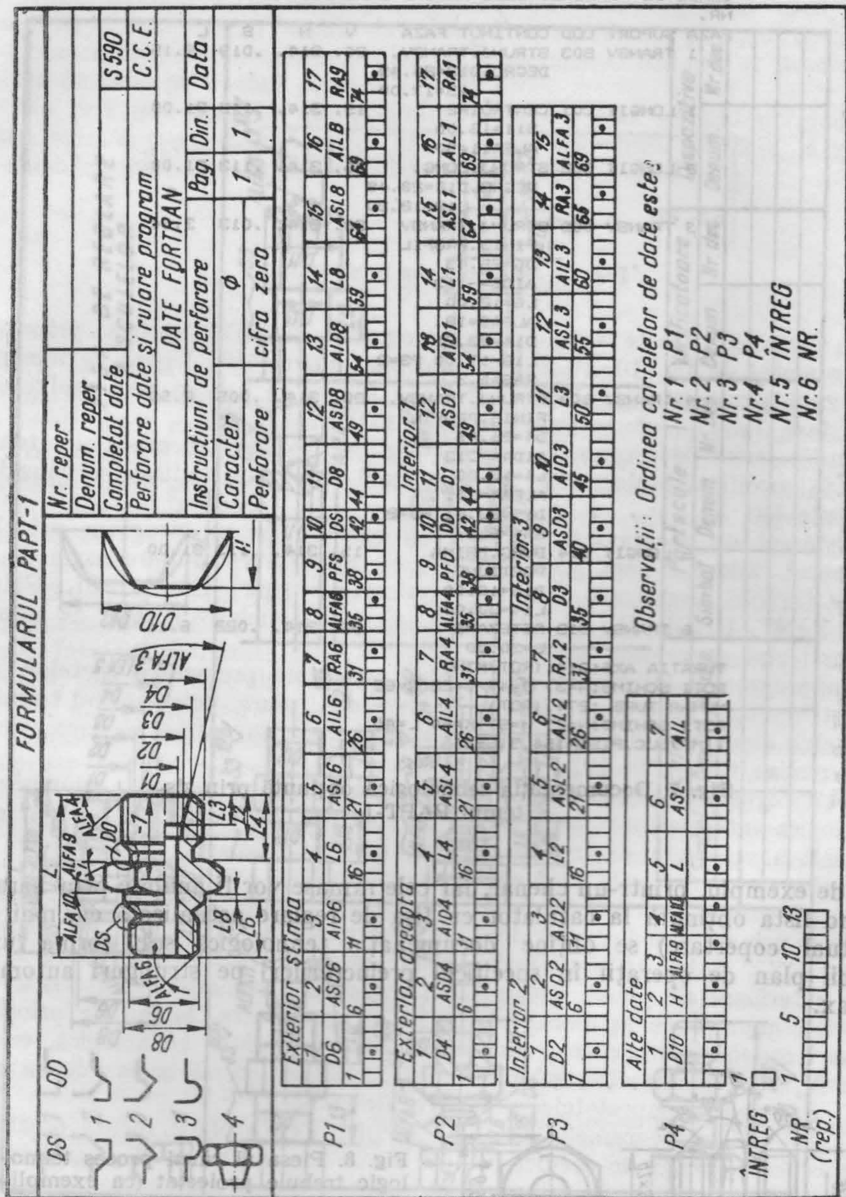


Fig. 6. Formularul PAPT-1.

```
     S I S T E M U L   P A P T - 1
STRUNG AUTOMAT CU 6 AXE - MODEL : 1A 240-6
SECTIA:X73          REPER NR.43-5731-0258
CONDITII  RACIRE: ULEI RACIRE STAS 2800-61
NR.
FAZA SUPORT COD CONTINUT FAZA     V    N     S   L
   1 TRANSV S03 STRUNJ.TRANSV.  36. 314. .019  5.15
                DEGRO.D13=28.49
                      L13=17.00
   1 LONGIT C01 CENTRUIRE       15. 314. .113 31.00
                 D11=15.00
                 ALFA11=90
   2 LONGIT S02 STRUNJ.LONG.    36. 314. .113 31.00
                 DEGRO.D12=28.49
                       L12=12.20
   3 TRANSV S05 STRUNJ.TRANSV   28. 314. .013  3.64
                 FINIS.PROFIL
                 D6=25.49
                 AID6=-.13
                 L6=12.00
                 ALFA6=18.
                 D14=23.21
                 L16=17.00 PS=2
                 RA6=6.3
   4 TRANSV S08 STRUNJ.TRANSV.  28. 314. .005  2.50
                FINIS.PROFIL
                D4=25.49
                AID4=-.13
                L4=12.00
                ALFA4=18.
                D15=23.21 PD=2
                RA4=6.3
   5 LONGIT B04 BURG.PRIMA      15. 314. .113 31.00
                PORTIUNE
                D16=15.00
                L14=30.00
   6 TRANSV S13 RETEZARE        36. 314. .022  6.11
                L=30.00
TURATIA AXE:314.(ROT/MIN)
ROTI SCHIMB:A=37 B=47 V=22 G=62
NUMAR TURE :274.(ROT)
ROTI SCHIMB:M=27 M=57 K=44 L=40
TIMP/BUC.PIESA:54.3(SEC)
```

Fig. 7. Documentația tehnologică obținută prin sistemul PAPT-1.

țiate, de exemplu, printr-un chenar, iar cele rămase vor fi anulate prin barare. Cuplînd lista obținută la caculator cu fișa de reglare actualizată ca mai sus (eventual copertare) se obține documenația tehnologică sub forma unui variabil plan de operații în specificul prelucrărilor pe strunguri automate multiax.

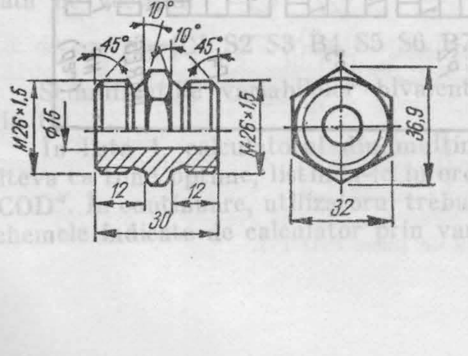


Fig. 8. Piesa al cărui proces tehnologic trebuie proiectat (ca exemplificare).

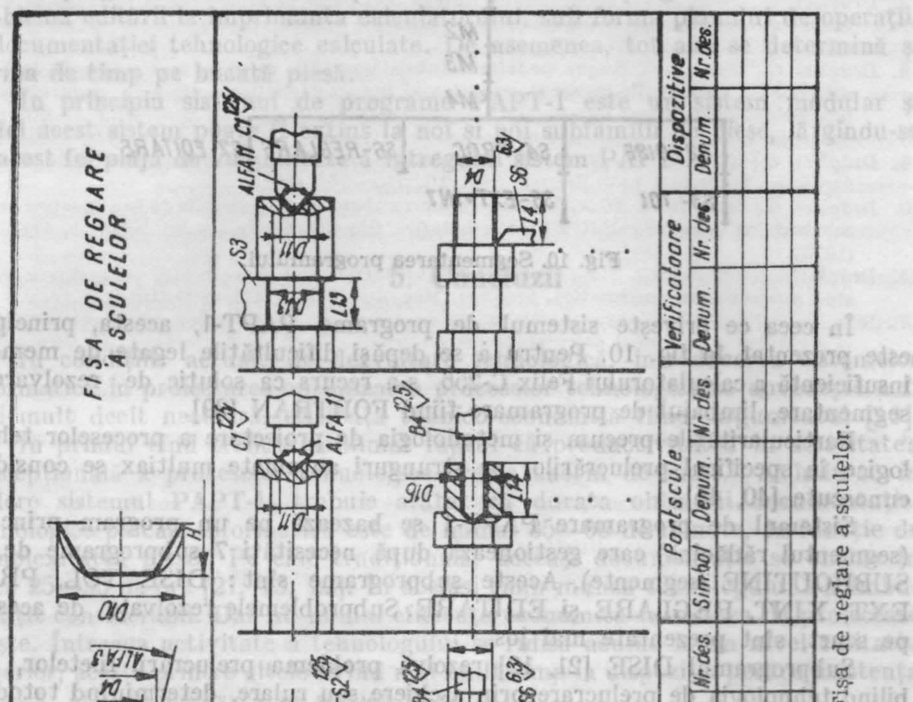


Fig. 9 Fișa de reglare a sculelor.

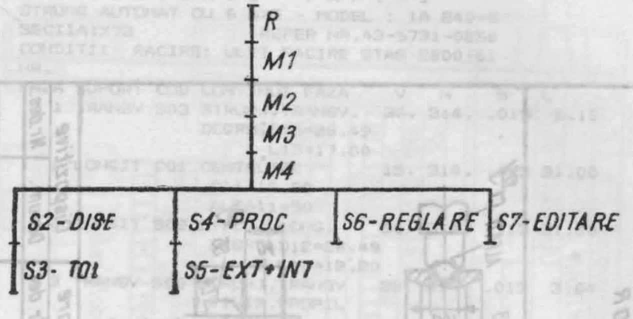


Fig. 10. Segmentarea programului.

În ceea ce priveşte sistemul de programe PAPT-1, acesta, principial, este prezentat în fig. 10. Pentru a se depăşi dificultăţile legate de memoria insuficientă a calculatorului Felix C-256, s-a recurs ca soluţie de rezolvare la segmentare, limbajul de programare fiind FORTRAN [39].

Particularităţile precum şi metodologia de proiectare a proceselor tehnologice în specificul prelucrărilor pe strunguri automate multiax se consideră cunoscute [40, 45].

Sistemul de programare PAPT-1 se bazează pe un program principal (segmentul rădăcină) care gestionează după necesităţi 7 subprograme de tip SUBROUTINE (segmente). Aceste subprograme sînt: DISE TOL, PROC, EXT, XINT, REGLARE şi EDITARE. Subproblemele rezolvate de acestea, pe scurt, sînt prezentate mai jos.

Subprogramul DISE [21, 12] rezolvă problema prelucrării filetelor, stabilind tehnologia de prelucrare prin aşchiere sau rulare, determinînd totodată parametrii dimensionali (cote nominale şi abateri) ai suprafeţei pe care ulterior se va realiza filetul [9, 10].

Subprogramul TOL calculează toleranţele optime pentru dimensiunile tehnologice intermediare în funcţie de procedeul aşchietor implicat şi cota nominală a suprafeţei ce trebuie prelucrată [21].

Subprogramul PROC [21] este acel subprogram care are ca sarcină stabilirea procedeelor optime de prelucrare. De fapt subprogamul PROC rezolvă problema prin intermediul celor două subporgrame EXT şi XINT [21]. Subprogramul EXT stabileşte procedeele optime de prelucrare pentru toate suprafeţele exterioare ale piesei, determinînd totodată parametrii caracteristici ca : factorii regimului de aşchiere, adaosurile de prelucrare, indicii de prioritate necesari ordonării procedeelor într-o anumită succesiune tehnologică, cursele de lucru, alte calcule etc. Subporgramul XINT realizează acelaşi lucru ca şi EXT, dar pentru suprafeţele interioare ale piesei.

Legat de subprogramele PROC, EXT şi XINT, trebuie făcută precizarea că procedeele optime de prelucrare sînt decise din mulţimea X a procedeelor avute iniţial în vedere în baza caracteristicilor geometrice şi de precizie ale piesei (date prin formularul PAPT-1). Alte criterii care trebuie avute în vedere la selectarea optimă a procedeelor au fost utilizate în faza de preoptimizare.

Subprogramul REGLARE [21] rezolvă problema punerii de acord a datelor calculate în cadrul celorlalte subprograme cu posibilităţile reale ale maşinii-

unelte implicate, stabilind şi roţile de schimb pentru reglarea lanţurilor cinematice.

Subprogramul EDITARE [21, 20, 22] este subprogramul care rezolvă problema editării la imprimanta calculatorului, sub forma planului de operaţii, a documentaţiei tehnologice calculate. De asemenea, tot aici se determină şi norma de timp pe bucată piesă.

În principiu sistemul de programe PAPT-1 este un sistem modular şi astfel acest sistem poate fi extins la noi şi noi subfamilii de piese, lărgindu-se în acest fel plaja de valabilitate a întregului sistem PAPT-1.

5. Concluzii

În condiţiile actuale de dezvoltare tehnologică, introducerea sistemelor informatice în proiectarea-optimizarea proceselor tehnologice se dovedeşte a fi mai mult decît necesară, eficienţa tehnico-economică fiind argumentul [16]. În primul rînd trebuie subliniat faptul că productivitatea în activitatea concepţională a proceselor tehnologice creşte enorm de mult. Avîndu-se în vedere sistemul PAPT-1, trebuie arătat că durata obţinerii documentaţiei tehnologice pe cale informatică este de numai 35—60 de minute, în funcţie de complexitatea piesei. Pe cale tradiţională, aceeaşi documentaţie se obţine în circa 25—30 de ore [21, 13, 22]. În acelaşi timp munca tehnologului s-a îmbunătăţit considerabil. Dar nu numai eficienţa economică în munca de proiectare creşte. Întreaga activitate a tehnologului se ridică acuma la un nivel calitativ superior, acesta printre altele avînd mai mult timp la dispoziţie pentru asistenţa tehnică, atît de necesară în atelier.

Paralel cu cele de mai sus, proiectarea asistată de calculator asigură optimizarea procesului tehnologic, cu toate consecinţele referitoare la costul şi productivitatea uzinării pieselor. Un alt mare avantaj al sistemului PAPT-1 este acela că nu necesită un personal utilizator specializat în calculatoare.

În încheiere, trebuie menţionat faptul că, prin prisma modului în care s-au pus şi s-au rezolvat problemele, sistemul PAPT-1 poate fi generalizat şi la prelucrările altor tipuri de piese.

BIBLIOGRAFIE

1. *Bjorke, O., Haugrud, B.*, **Mathematical Methode in Planning of Machining Operations.** Annals of the C.I.R.P., XVI, 4, 399—407 (1968).
2. *Carro, Cao, G.*, **Considerations sur le système machine-pièce-outil à l'egard de l'optimalisations des usinages par enlèvement de eopeau.** Annals of the C.I.R.P., XVIII, 4, 633—652 (1970).
3. *Chiriacescu, T.S.*, **Automatizarea proceselor tehnologice.** Universitatea din Braşov, 1975.
4. *Dragomirescu, M., Maliţa, M.*, **Programarea pătratică.** Ed. ştiinţifică, Bucureşti, 1968.
5. *Drăghici, G., Ivan, N.V.*, **Über die automatische Entwerfung der technologischen Vorgänge mit Verwendung von elektronischen Rechenmaschinen.** Vorträge „COMPCONTROL '72" — Anwendung der Elektronenrechner in der Leitung der Industrieunternehmen, III. Section, Sopron, 115—130 (1972).

6. *Drăghici, G., Ivan, N.V. und Iacobs, H.J.*, **Die Boolsche Algebra als Hilfsmittel zur optimalen Gestaltung diskreter technologischer Prozesse der mechanischen Teilfertigung (Prozessoptimierung).** Wiss. Z. Techn. Universität Dresden, 23, 2, 385—391 (1974).
7. *Drăghici, G., Ivan, N.V.*, **Programarea liniară mai bună ca programarea convexă în unele cazuri de determinare a parametrilor optimi de exploatare a sculelor.** Construcţia de maşini, XXIX, 12, 598—601, (1977).
8. *Drăghici, G., Ivan, N.V.*, **Despre modelarea subsistemului de calcul al regimurilor de aşchiere utilizate la strungurile automate multiax şi luarea deciziilor optime cu ajutorul calculatoarelor electronice.** Comunicările conferinţei din domeniul proceselor şi utilajelor de prelucrări la rece. Institutul politehnic „Traian Vuia" Timişoara, 49—58 (1970).
9. *Drăghici, G, Ivan, N.V., Stroe, M.*, **Berechnung des Ausgangsdurchmessers beim Gewinderollen.** Werkstatt und Betrieb, 104, 10, 755—758 (1971).
10. *Drăghici, G., Ivan, N.V., Stroe, M.*, **Determinarea abaterilor admisibile de la diametrul semifabricatelor în cazul rulării filetelor metalice.** Construcţia de maşini, XXV, 8, 471—474 (1973).
11. *Drăghici, G., Chiriacescu, T.S.*, **Calculul durabilităţii sculei şi regimului de aşchiere folosind programarea matematică.** St. Cerc. Mec. Apl., 30, 4, 975—999 (1971).
12. *Ivan, N.V.*, **Contribuţii asupra sistemelor de concepţie a proceselor tehnologice. Modele şi rezolvări cu ajutorul calculatoarelor electronice.** Teză de doctorat. Universitatea din Braşov, 1977.
13. *Ivan, N.V.*, **Bazele optimizării proceselor tehnologice în construcţia de maşini.** Curs, Universitatea din Braşov, 1983.
14. *Ivan, N.V*, **Calculul regimurilor optime de aşchiere prin programare convexă.** Construcţia de maşini, XXV, 3, 151—156 (1973).
15. *Ivan, N.V.*, **Programarea algoritmului SIMPLEX pe calculatoare electronice, în vederea calculului regimurilor optime de aşchiere.** Buletinul Universităţii din Braşov, A, XIV, 407—416 (1972).
16. *Ivan, N.V.*, **Asupra rentabilităţii utilizării calculatoarelor în proiectarea proceselor tehnologice.** Noutăţi în mecanica aplicativă şi în construcţia de maşini. Universitatea din Braşov, XVII, B, 87—94 (1975).
17. *Ivan, N.V.*, **Un sistem informatic de proiectare a proceselor tehnologice de prelucrare pe maşinile-unelte aşchietoare.** Buletinul Universităţii din Braşov, A, XX, C, 127—132 (1978).
18. *Ivan, N.V.*, **Proiectarea asistată de calculator a adaosurilor optime de prelucrare şi dimensiunilor tehnologice intermediare.** A IV-a Conferinţă de procese şi utilaje de prelucrare la rece. Institutul politehnic „Traian Vuia" Timişoara, 1, 277—282 (1981).
19. *Ivan, N.V.*, **Modelarea matematică a datelor experimentale necesare rezolvării prin calculator a problemei adaosurilor de prelucrare.** Construcţia de maşini, XXIV, 2, 91—94 (1982).
20. *Ivan, N.V.*, **Subprogram de editare a operaţiilor şi/sau fazelor în ordinea succesiunii tehnologice optime.** Lucrări ştiinţifice, 3, B, 147—152 (1981), Institutul de învăţămînt superior Tg. Mureş.
21. *Ivan, N.V.*, **Verificarea în producţie a proceselor tehnologice proiectate cu ajutorul calculatoarelor electronice.** Universitatea din Braşov, 1979.
22. *Ivan, N.V.*, **Bazele optimizării proceselor tehnologice.** Îndrumar de laborator. Universitatea din Braşov, 1982.
23. *Ivan, N.V.*, **Viteza de avans, parametru important de optimizare în unele cazuri de prelucrare cu multiscule.** Buletinul Universităţii din Braşov, XXIII, A, 255—257 (1981).
24. *Ivan, N.V., Boncoi, Gh.*, **Privitor la calculul regimurilor de aşchiere pe strunguri automate multiax.** Buletinul Institutului politehnic Braşov. A, XII, 427—438 (1970).
25. *Ivan, N.V., Tănase, E.*, **Determinarea parametrilor regimurilor de aşchiere utilizate la strunguri automate.** Construcţia de maşini, XXVII, nr. 2—3, 129—131 (1975).
26. *Ivan, N.V., Drăghici, G.*, **PAPT-1, un sistem de proiectare automată a proceselor tehnologice optime de prelucrare pe strunguri automate multiax.** A II-a Conferinţă Naţională de Maşini-unelte, decembrie, Bucureşti, 82—89 (1976).
27. *Ivan, N.V., Drăghici, G.*, **Privitor la asigurarea unei exploatări optime a sculelor prin proiectarea automată a proceselor tehnologice.** Noutăţi în proiectarea, tehnologia şi exploatarea sculelor aşchietoare. Universitatea din Braşov, 209—214 (1977).
28. *Ivan, N.V., Stratulat, P.*, **Tipizarea, mijloc de optimizare prin calculator a proceselor tehnologice.** A III-a Conferinţă de maşini-unelte şi scule, mai, Bucureşti, 51—55, (1979).

29. *Ivan, N.V., Ivan, M.C., Tiței, D.*, **Practica proiectării asistate de calculator a proceselor tehnologice de prelucrare pe strunguri automate multiax.** A 3-a sesiunea de comunicări științifice „Creația tehnică și fiabilitatea în construcția de mașini". Probleme tehnologice de prelucrări mecanice în construcția de mașini. Institutul politehnic „Gh. Asachi", Iași, 164—170 (1981).

30. *Ivan, N.V., Tiței, D., Ivan, M.C.*, **Program operativ de optimizare a regimului de așchiere la prelucrările cu multiburghie.** Lucrări științifice, 3, B, 141—146 (1981). Institutul de învățămînt superior, Tg. Mureș.

31. *Ivan, N.V., Drăghici, G.*, **Proiectarea automată a proceselor tehnologice optime în cazul prelucrărilor pe strunguri automate multiax.** Universitatea din Brașov, 1976.

32. *Ivan, N.V., Leu, A., Groos, A., Cziszer, L.*, **Modele matematice pentru calculul adaosurilor de prelucrare.** Buletinul Institutului politehnic „Gh. Asachi" Iași, 1984.

33. *Ivănescu, P.L.* **Programarea pseudo-booleeană și aplicații ale ei.** Studii și cercetări matematice, 17, 4, 583—598 (1975).

34. *Ivănescu, P.L., Rudeanu, S.*, **O teorie a deciziilor binare. Cazul liniar.** Studii și cercetări de calcul economic și cibernetica economică, 2, 105—127 (1966).

35. *Ivănescu, P.L. Rosenberg, I., Rudeanu, S.*, **Asupra determinării minimelor funcțiilor pseudo-booleene.** Studii și cercetări matematice, 14, 359—364 (1963).

36. *Jacobs, H.J., Jacob, E., Kochan, D.*, **Spannungsoptimierung.** VEB Verlag Technik, Berlin, 1977.

37. *Kovan, V.M., ș.a.* **Pravocinik tehnologo mașino-stroiteliia,** 1. Mașghiz, Moskva, 1963.

38. *Mihăilă, M.*, **Introducere în programarea liniară.** Ed. didactică și pedagogică, București, 1970.

39. *Niculescu, Șt.*, **Inițiere în FORTRAN.** Ed. tehnică, București, 1972.

40. *Petriceanu, Gh., Gyenge, Cs., Morar, L.*, **Proiectarea proceselor tehnologice și reglarea strungurilor automate.** Ed. Tehnică, București, 1979.

41. *Picos, C., Ailincăi, Gh., Bohosievici, C., Pruteanu, O., Coman, Gh., Brana, V., Paraschiv, D.*, **Calculul adaosurilor de prelucrare și al regimurilor de așchiere.** Ed. Tehnică, București, 1974.

42. *Pisău, Gh., Toma, C., Mihăescu, I.*, **Elaborarea și introducerea sistemelor informatice.** Ed. Tehnică, București, 1975.

43. *Rumsiski, L.Z.*, **Prelucrarea matematică a datelor experimentale. Îndumar.** Ed. Tehnică, București, 1974 (traducere din limba rusă).

44. *Savas, E.S.* **Conducerea cu calculatoare a proceselor industriale.** Ed. Tehnică, București, 1969 (traducere din limba engleză — S.U.A.).

45. *Vlad, A., Grigorescu, N.*, **Reglarea strungurilor automate.** Ed. Tehnică, București, 1965.

CONTINUARE DE LA PAGINA 16

```
186   TYPE OPERATOR=PROCESS(ACCES:ACCES_BANDA;I:INTEGER);
187   [#############################################]
188   VAR T:IE_TERMINAL;LIN:LINIE;DOC:DOCUMENT;
189   PROCEDURE CIT(MES:LINIE;VAR LIN:LINIE);
190   BEGIN T.SCRMESAJ(MES); T.CITMESAJ(LIN) END; [SCRIERE CU RASPUNS]
191   PROCEDURE CITDOCUMENT;
192   BEGIN
193      WITH DOC DO
194         BEGIN
195         CIT('CIMP1(:0:)',C1);
196         CIT('CIMP2(:0:)',C2);
197         CIT('CIMP3(:0:)',C3)
198         END
199   END;
200   BEGIN
201      INIT T(CHR(I+ORD('0')));
202      CYCLE
203         T.ASTBRK;
204         CITDOCUMENT;
205         ACCES.SCRARTICOL(DOC,I);
206         T.SCRMESAJ('DUCUMENT ACCEPTAT(:13:)(:21:)(:0:)');
207      END;
208   END;
209   [####################]
210   [ PROCESUL INITIAL ]
211   [####################]
212   VAR M:ACCES_BANDA;
213      CENTR:CENTRAL;
214      OPER:ARRAY(.1..NRPROCESE.) OF OPERATOR;
215      I:INTEGER;
216   BEGIN
217      INIT M,CENTR(M);
218      FOR I:=1 TO NRPROCESE DO INIT OPER(.I.)(M,I);
219   END.
```

Fig. 4. (Programul 2). Exemplu de program conversațional în limbajul PASCAL CONCURENT.

3. Programe concurente. Sincronizarea și comunicarea între procese

3.1. Procese concurente

Marea majoritate a programelor realizate în limbajele de nivel înalt tradiționale (FORTRAN, COBOL, PASCAL etc.) sînt concepute de așa natură, încît acțiunile lor se desfășoară în mod strict secvențial (fiecare acțiune este inițiată doar cînd predecesoarea ei s-a terminat). Un astfel de program reprezintă un proces.

Uneori devine însă necesară conceperea unor programe care presupun executarea în paralel a mai multor procese. Aceasta conduce pe de o parte la exploatarea eficientă a resurselor sistemului de calcul (sisteme cu multiprogramare), iar pe de altă parte la realizarea unei analogii cu însăși natura problemei de rezolvat (programe pentru conducerea proceselor, programe conversaționale, programarea sistemelor multiprocesor și a sistemelor distribuite etc.) Deseori procesele paralele gestionează în comun resurse ale sistemului și între ele pot exista relații de cooperare (schimb de mesaje, transfer de date) sau conflictuale (prin solicitarea aceleiași resurse în același timp). Astfel de procese se numesc *concurente*.

Coexistența proceselor concurente care în marea majoritate a timpului se desfășoară asincron, impune în anumite momente sincronizarea în vederea realizării unei comunicări. Relațiile între ele se pot reduce în final la următoarele două situații tipice:

a) *Excluderea mutuală* între procese care folosesc în comun aceeaşi resursă, la care accesul este permis doar dintr-un singur proces la un moment dat. În acest caz este necesară secvenţializarea cererilor de acces la resursa respectivă.

b) *Cooperarea* între procese care schimbă mesaje sau în general date cel mai des în relaţia producător/consumator. Informaţiile produse de un proces sînt utilizate de celălalt ; schimbul de mesaje presupune în anumite situaţii sincronizarea lor.

Spre deosebire de programarea secvenţială, care permite definirea unui singur proces izolat, descrierea unor procese concurente şi a relaţiilor dintre ele se face prin aşa-numita *programare concurentă*.

Limbajul PASCAL CONCURENT [2] oferă mijloace de rezolvare într-un limbaj de nivel înalt a categoriei cu totul noi de probleme puse de programarea concurentă.

Respectînd întru totul principiile şi forma limbajului PASCAL, s-au introdus cîteva concepte noi, cum ar fi tipurile sistem, sincronizarea programată şi automată, timpul real, intrări/ieşiri la nivel fizic [2, 7, 9] etc.

Programul 2 (fig. 4) este un program concurent care permite introducerea în paralel de la mai multe terminale a unor înregistrări, concentrate în final pe o bandă magnetică. Un operator central are posibilitatea de a porni şi opri sesiunea de lucru la fiecare terminal, respectiv de a încheia lucrul la toate terminalele, după care închide fişierul creat. În paragrafele următoare se vor face referiri repetate la acest program.

3.2. Procesele — elemente active ale programului concurent

În PASCAL CONCURENT procesele unui program sînt variabile de tip *process*. Acţiunile descrise în corpul de instrucţiuni din blocul fiecărui proces reprezintă activităţile paralele ce au loc în program. Datele declarate într-un proces sînt proprii acestuia şi sînt inaccesibile din alte componente sistem. Procese concurente pot gestiona în comun resurse prin intermediul altor componente de tip sistem, la care au drept de acces. Acestea se descriu ca parametri formali ai tipului *process* şi se explicitează ca argumente la lansarea (iniţializarea) procesului.

În programul 2 tipul proces „central" descrie activitatea operatorului central (liniile 153—183). Tipul proces „operator" descrie activitatea operatorului de la terminal (185—207).

Din listele de parametri ale tipurilor proces „central" şi „operator" rezultă că ambele au acces la un monitor de tip „acces-banda".

3.3. Monitoarele — elemente de sincronizare şi comunicare. Sincronizarea automată

Descrierea unor zone de date folosite în comun de mai multe procese şi a tuturor operaţiilor permise asupra acestora se face printr-o definiţie de tip *monitor*. Procesele nu pot acţiona direct asupra datelor declarate în monitor. Acestea sînt accesibile din exterior doar în mod indirect, prin intermediul unor rutine externe ale monitorului.

Datele dintr-o variabilă de tip monitor reprezintă deci resurse partajate între procese. Protecţia acestora faţă de accesul simultan din mai multe procese (excluderea mutuală) se rezolvă în mod automat, după cum urmează :

— un proces solicită acces la un monitor apelînd una din rutinele acestuia ;

— pe parcursul executării rutinei apelate, procesul dobîndeşte acces exclusiv la monitor, care devine astfel ocupat ;

— la terminarea rutinei apelate, monitorul se eliberează ;

— cererile de acces la un monitor ocupat au ca rezultat punerea în aşteptare a proceselor respective ; cererile se achită pe rînd, la eliberarea monitorului de către procesul precedent.

Prin acest mecanism, transparent pentru programator, se realizează sincronizarea automată a proceselor.

În programul 2, prin intermediul monitorului de tip „acces-banda" se realizează accesul proceselor la banda magnetică.

3.4. Clase

O clasă are rolul de a descrie o structură de date şi accesul controlat, prin intermediul unor operaţii precizate (rutinele clasei) la aceste date. Referirile din afara clasei la datele şi rutinele ei pot avea loc, prin definiţie, doar dintr-o singură componentă sistem (spre deosebire de monitoare). Aceasta se verifică la compilarea programului.

Descriind în mod unitar date şi acţiuni, clasa este un mijloc de modularizare şi structurare a programului, fără a implica aspecte privind concurenţa şi sincronizarea în cadrul sistemului concurent (cum este cazul proceselor şi monitoarelor).

În programul 2 tipurile clasă „ie-terminal" (liniile 22—75), „ie-bandă" (77—94), „ie-consola" (96—127) furnizează proceduri pentru introducerea şi extragerea de mesaje la terminal respectiv consolă, pentru scrierea pe bandă şi închiderea fişierului, pentru punerea în aşteptare a procesului „central" pînă la intervenţia operatorului etc. [9].

3.5. Componentele programului concurent. Drepturi de acces

Tipurile sistem introduse mai sus sînt după cum s-a arătat tipuri abstracte cuprinzînd descrierea unor date şi a unor operaţii posibile. Componentele efective ale programului concurent, procese, monitoare, şi clase sînt variabile de aceste tipuri. În cazul în care în structura programului intră mai multe componente definind operaţii identice asupra unor date de aceeaşi structură, acestea se vor declara ca variabile de acelaşi tip sistem.

În prog. 2 s-a declarat un tablou de procese „oper" cu „nrprocese" elemente de tip „operator" (linia 213), care se pun în lucru (se iniţializează) pe linia 217. În continuare, activităţile acestor procese se vor desfăşura în paralel.

O componentă sistem are acces la parametrii şi variabilele ei proprii. De asemenea, componenta poate avea acces la alte variabile de tip sistem explicitate ca argumente la iniţializarea ei.

Astfel, la definirea tipurilor proces din programul 2 se indică accesul la un monitor de tip „acces-banda“ (liniile 153 şi 185). La iniţializarea variabilelor proces respective (liniile 216 şi 217) se concretizează monitorul (variabila „m“).

Parametrii formali de tip sistem dintr-o definiţie de tip *process* sau *monitor* pot fi doar monitoare. Astfel, resursele partajate între procese sînt gestionate exclusiv prin monitoare.

Accesul din exterior la o componentă sistem se face apelînd rutine ale ei. Apelul rutinei se realizează prin numele acesteia care se va califica (v. § 2.2.3.2) cu identificatorul componentei din care face parte.

Reprezentarea grafică a componentelor unui program concurent şi a legăturilor dintre ele se poate face cu ajutorul unui graf de acces. Componentele sistemului se reprezintă prin cercuri, iar drepturile de acces (legăturile dintre componente) prin săgeţi. Un graf de acces este prin urmare un graf orientat, organizat ierarhic pe baza drepturilor de acces ale componentelor sistemului. El nu poate conţine circuite, evitîndu-se astfel blocările datorate unor apelări recursive ale monitoarelor. Respectarea organizării în acest mod a drepturilor de acces se verifică riguros la compilarea programului. În fig. 5 se prezintă graful de acces corespunzător programului 2.

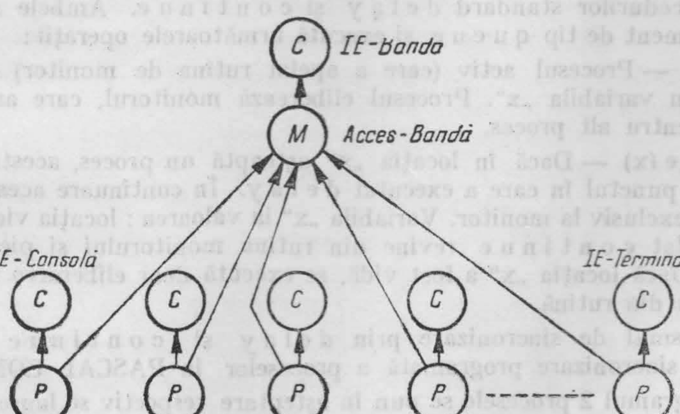


Fig. 5. Graful de acces pentru programul 2

Fiecare proces de tip „operator“ are acces la o componentă de tip „ie-terminal“, declarată ca variabilă în interiorul procesului (linia 187). În mod asemănător tipul proces „central conţine o variabilă (clasă) de tip „ie-consola“ iar tipul monitor „acces-banda“ conţine o clasă de tip „ie-banda“. Toate procesele au acces la monitorul de tip „acces-banda“.

3.6. Sincronizarea programată a proceselor

Sincronizarea automată realizată prin excluderea mutuală a apelurilor procedurilor de monitor, nu permite rezolvarea tuturor problemelor care apar la interacţiunea proceselor.

O astfel de situaţie apare spre exemplu în cazul a două procese p1 şi p2 care cooperează prin intermediul unui tampon T, de dimensiune finită. Procesul p1 (producătorul) introduce articole în tampon, iar p2 (consumatorul) le extrage. Cele două procese funcţionează în paralel, asincron şi independent unul de celălalt. Se pot ivi însă două situaţii limită în care tamponul s-a golit sau a devenit plin. Astfel, devine necesar ca procesul p1 să fie pus în aşteptare prin program cînd găseşte tamponul plin ; în mod analog este necesar ca p2 să fie pus în aşteptare cînd găseşte tamponul gol. După modificarea stării tamponului, care a necesitat punerea în aşteptare, tot prin program procesele trebuie relansate. S-a ilustrat astfel, într-o situaţie tipică, necesitatea ca programatorul, în funcţie de anumite condiţii, să pună în aşteptare un proces care apoi să poată fi relansat tot din program.

Procesele se pun în aşteptare în variabile de tipul standard q u e u e. Acestea reprezintă locaţii de aşteptare care, la un moment dat, pot conţine un singur proces.

Variabile de tip q u e u e, sau structuri cu componente de acest tip, se declară doar în monitoare. Prin urmare toate acţiunile de punere în aşteptare şi relansare a proceselor au loc în monitoare. Ele se realizează prin intermediul procedurilor standard d e l a y şi c o n t i n u e. Ambele se apelează cu un argument de tip q u e u e şi execută următoarele operaţii :

d e l a y (x) — Procesul activ (care a apelat rutina de monitor) se pune în aşteptare în variabila „x". Procesul eliberează monitorul, care astfel devine accesibil pentru alt proces.

c o n t i n u e (x) — Dacă în locaţia „x" aşteaptă un proces, acesta se reactivează din punctul în care a executat d e l a y. În continuare acest proces va avea acces exclusiv la monitor. Variabila „x" ia valoarea : locaţia vidă. Procesul care a apelat c o n t i n u e revine din rutina monitorului şi pierde accesul la acesta. Dacă locaţia „x" a fost vidă, se execută doar eliberarea monitorului şi revenirea din rutină.

Mecanismul de sincronizare prin d e l a y şi c o n t i n u e reprezintă tehnica de sincronizare programată a proceselor în PASCAL CONCURENT.

În programul 2 procesele se pun în aşteptare respectiv se lansează la cererea operatorului central. Aceste operaţii se realizează în monitorul de tip „acces-banda" ca urmare a unor apeluri din procesul de tip „central" (liniile 178, 179, 180). Punerea în aşteptare a procesului cu indice „i" se face în locaţia corespunzătoare din tabloul „locaţii", cu elemente de tip q u e u e (linia 132).

4. Concluzii

S-au prezentat cîteva aspecte fundamentale privind programarea structurată şi concurentă şi două limbaje, PASCAL şi PASCAL CONCURENT, care facilitează în mare măsură aplicarea în practică a acestor tehnici. Aceste limbaje sînt accesibile programatorului din ţara noastră nu numai pe calculatoarele din familia Felix C, ci şi pe calculatoarele Felix M-18, Independent şi Coral.

BIBLIOGRAFIE

1. *Wirth, N.* **The Programming Language Pascal**, Acta Informatica, nr. 1, 1971.
2. *Hansen, P.B.* **The Programming Language Concurrent Pascal**, IEEE Transaction on Softvare Engineering, vol. 1, nr. 2, 1975.
3. *Jitaru, M.* **Îndrumător de limbaje de programare**, Ed. Tehnică, București, 1978.
4. *Văduva, I.* **Programarea structurată**, Ed. Tehnică, București 1978.
5. *Niculescu, S* **Fortran – inițiere în programarea structurată**, Ed. tehnică, București, 1979.
6. * * * **Programarea și utilizarea calculatoarelor – studii și aplicații**, Ed. FACLA, 1981.
7. * * * **Limbajul Pascal**. Manual de utilizare, CCE al IPT, 1981.
8. * * * **Limbajul Pascal**. Manual de operare, CCE al IPT, 1982.
9. * * * **Limbajul Pascal Concurent**. Manual de utilizare, CCE al IPT, 1980.
10. * * * **Limbajele Pascal și Pascal Concurent, V2, Manual de utilizare** și operare, CCE al IPT, 1983.
11. *Dahl, O.J.* **Structured Programing**, Academic Press London, New York, 1972.
12. *Niculescu, S.* **Macroprocesoare și limbaje extensibile**, Editura științifică și enciclopedică, București, 1982.

VĂ RECOMANDĂM :

Din seria Automatică-Management-Calculatoare (AMC) apar în trimestrul III 1985 volumele : AMC 48, AMC 49, AMC 50, AMC 51, în cuprinsul cărora sînt prezente module și cicluri de foarte mare actualitate, de înaltă calitate și de un interes deosebit, și anume :

— Congresul mondial trienal al Federației Internaționale de Automatizare (IFAC) „O punte între știință și tehnologie", Budapesta 1984, reprezentat prin plenare, studii de caz și sinteze pentru toate secțiunile (autori străini și români).

— „Societatea informatică" note de lectură după cartea japonezului Masuda, „Resursele informaționale naționale" și „Fenomenul calculatoarelor personale" după sovieticul Gromov.

— „Memento de teleprelucrare", cu toate informațiile necesare pentru echipamentele și sistemele teleinformatice românești.

— „Minicalculatoarele INDEPENDENT și CORAL". Manual de utilizare din ciclul SERVICE pentru CALCULATOARE.

— „BASIC pentru începători, cu calculatorul personal", un manual practic din ciclul „CALCULATOARE PERSONALE ȘI PROGRAMAREA LOR".

— Ciclul „PROIECTAREA ASISTATĂ DE CALCULATOR" reprezentat prin articole de direcționare în domeniu și articole prezentate la o primă sesiune națională.

— Microcalculatoarele personale românești, Student-HC 80, PRAE (pentru acesta și limbajul său BASIC) și microcalculatorul profesional-personal românesc Felix PC, în prezentări sintetice — în premieră într-o carte.

În trimestrul IV 1985 apar și volumele AMC 52—53—54, cu ciclurile amintite, dar și cu automatizarea flexibilă, roboții, limbajul BASIC pentru WANG VS, ghidul analistului (continuare la AMC 45—46), jocuri de întreprindere ș.a.

Prețul unui volum AMC este de aproximativ 25 lei.

- Seria AMC se procură din librăriile cu carte tehnică.
- Din cursul anului 1984, de la volumul AMC 38, pe coperți apar cele cîteva domenii atacate, volumele căpătînd, astfel, titluri. Această tematizare s-a bucurat de succes nu numai în rîndul informaticienilor ci și al altor specialiști, ca și a unor cercuri de studenți, elevi ș.a. Cicluri continue ca : proiectarea asistată de calculator, analiza sistemelor, calculatoare personale, rețele de calculatoare, note de lectură, service pentru calculatoare, congrese ș.a., realizate de specialiști, atrag noi cititori permanenți.
- Pentru a vă asigura obținerea seriei AMC în noile condiții, recomandăm întreprinderilor să ne trimită pe adresa : EDITURA TEHNICĂ, Piața SCÎNTEII 1, comenzi ferme anticipate, semnate de director și contabil șef, care să colecteze atît volumele necesare documentării unității cît și cererile ferme exprimate de oamenii muncii prin organe sindicale (FUS, CIT ș.a.). Și cititorii individuali ne pot scrie, indicînd adresa exactă. Noi înaintăm toate aceste comenzi, cu indicații de prioritate, centrelor de librării, prin Întreprinderea de difuzare a cărții (I.D.C.).
- Plata se face la primirea volumelor, nu anticipat.
- In 1984 au apărut volumele 35–44, iar în 1985 (trim. I–II) AMC 45–47. În trim. III apar AMC 48–51, iar în trim. IV, AMC 52–54. Prețul unui volum este de 22–25 lei.
- In 1986 sînt prevăzute AMC 55–68.
- Eventualele lămuriri la tel. 18.06.30 sau 17.60.10/2100.

EDITURA TEHNICĂ

- Volum realizat prin cooperare cu Comisia de cibernetică a Academiei R. S. România, Institutul de tehnică de calcul și informatică, Institutul de proiectări automatizări, Întreprinderea pentru întreținerea și repararea utilajelor de calcul și cu specialiști din aceste foruri și unități.
- Cu sprijinul Federației Internaționale de Automatizare (IFAC) și cu concursul unor specialiști din R. P. Ungară.
- Cu concursul și al unor specialiști de la Institutul Politehnic București, Universitatea Brașov, Institutul Politehnic Timișoara, Institutul de Meteorologie și Hidrologie, Institutul de Cercetări și Proiectări pentru Gospodărirea Apelor.

lei 27